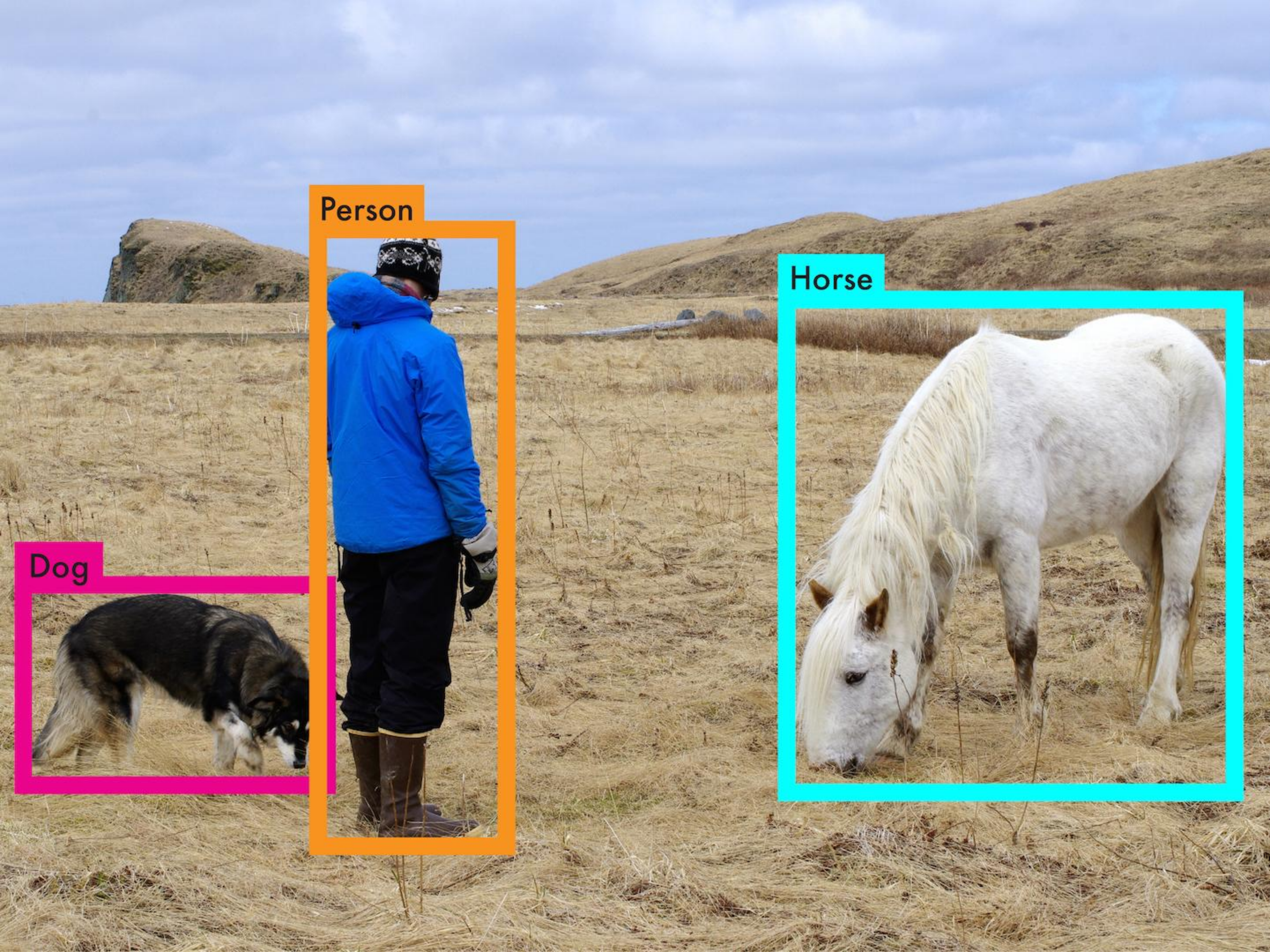

JOSEPH REDMON
ROSS GIRSHICK
SANTOSH DIVVALA
ALI FARHADI
Dog
MOST ACCURATE REAL-TIME DETECTOR 2016
FASTEST OBJECT DETECTOR IN THE LITERATURE 2016
"YOU ONLY LOOK ONCE"
REAL-TIME DETECTION


Person
Horse
Dog

Accurate object detection is slow!

	Pascal 2007 mAP	Speed	
DPM v5	33.7	.07 FPS	14 s/img

Accurate object detection is slow!

	Pascal 2007 mAP	Speed	
DPM v5	33.7	.07 FPS	14 s/img
R-CNN	66.0	.05 FPS	20 s/img

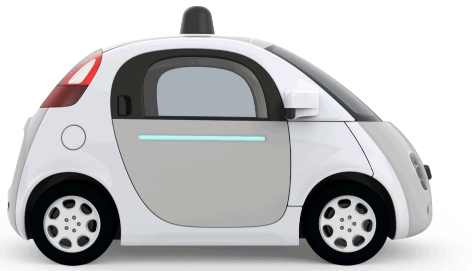
Accurate object detection is slow!

	Pascal 2007 mAP	**Speed**	
DPM v5	33.7	.07 FPS	14 s/img
R-CNN	66.0	.05 FPS	20 s/img

⅓ Mile, 1760 feet

Accurate object detection is slow!

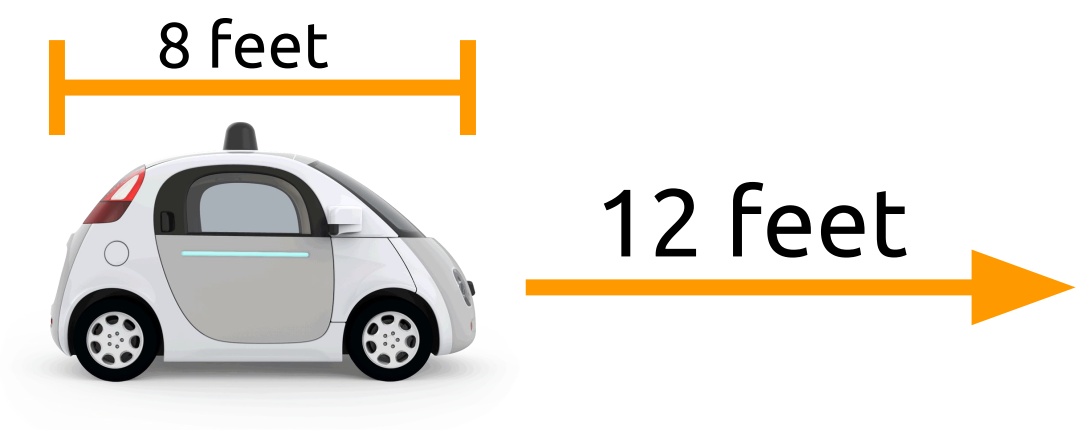
	Pascal 2007 mAP	Speed	
DPM v5	33.7	.07 FPS	14 s/img
R-CNN	66.0	.05 FPS	20 s/img
Fast R-CNN	70.0	.5 FPS	2 s/img



176 feet

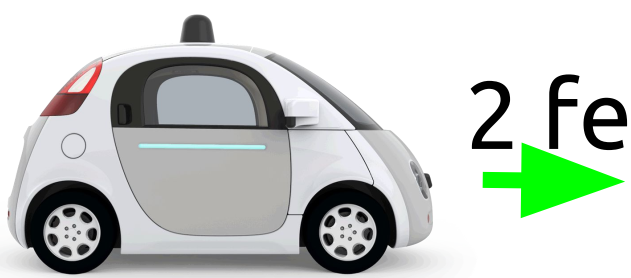
Accurate object detection is slow!

	Pascal 2007 mAP	Speed	
DPM v5	33.7	.07 FPS	14 s/img
R-CNN	66.0	.05 FPS	20 s/img
Fast R-CNN	70.0	.5 FPS	2 s/img
Faster R-CNN	73.2	7 FPS	140 ms/img



Accurate object detection is slow!

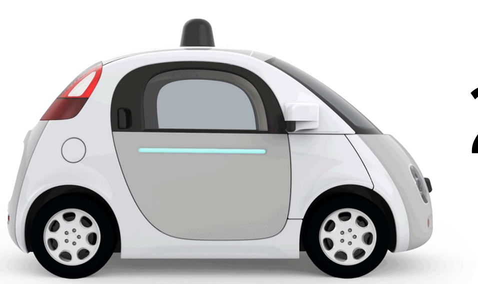
	Pascal 2007 mAP	Speed	
DPM v5	33.7	.07 FPS	14 s/img
R-CNN	66.0	.05 FPS	20 s/img
Fast R-CNN	70.0	.5 FPS	2 s/img
Faster R-CNN	73.2	7 FPS	140 ms/img
YOLO	63.4	45 FPS	22 ms/img



2 feet

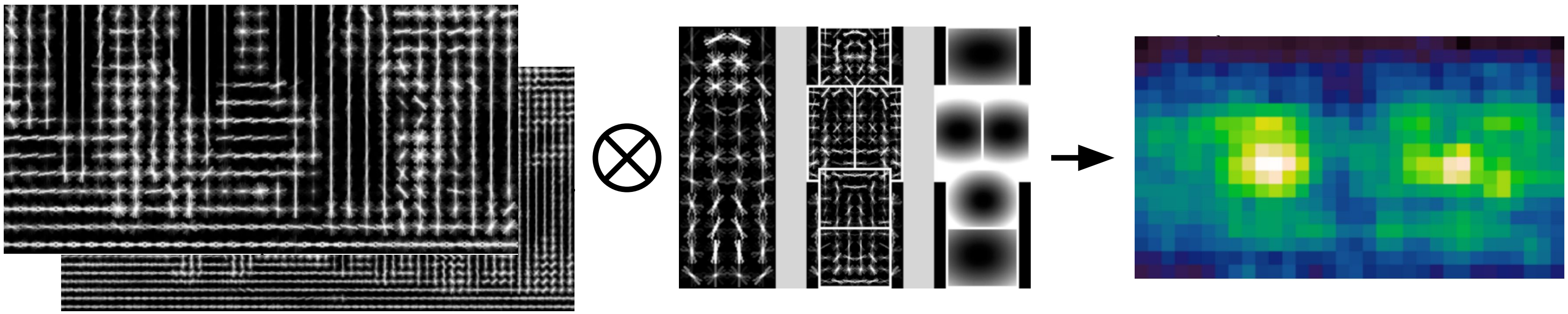
Accurate object detection is slow!

	Pascal 2007 mAP	Speed	
DPM v5	33.7	.07 FPS	14 s/img
R-CNN	66.0	.05 FPS	20 s/img
Fast R-CNN	70.0	.5 FPS	2 s/img
Faster R-CNN	73.2	7 FPS	140 ms/img
YOLO	~~63.4~~ 69.0	45 FPS	22 ms/img

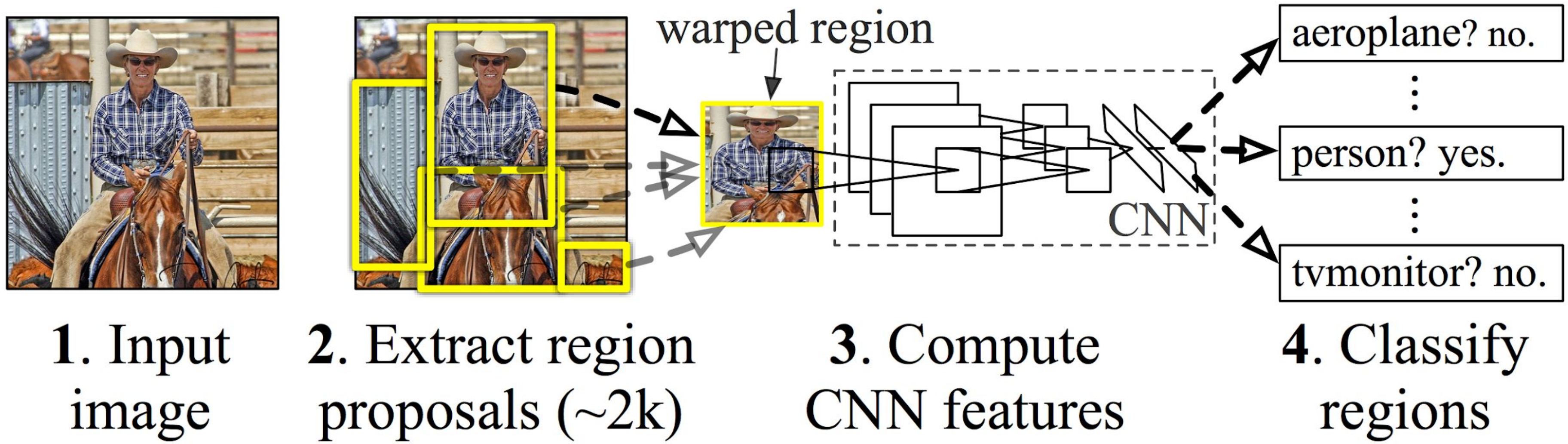


2 feet

DPM: *Deformable Part Models*

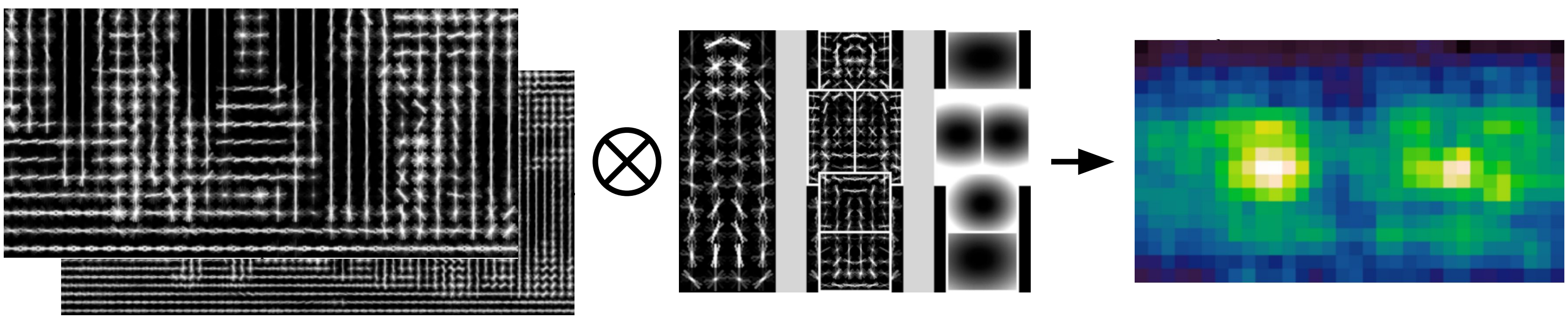


R-CNN: *Regions with CNN features*

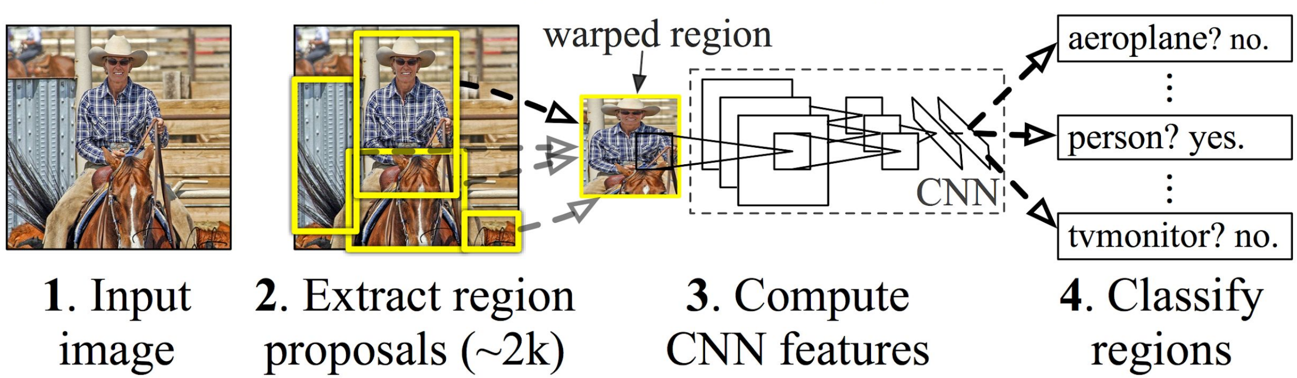


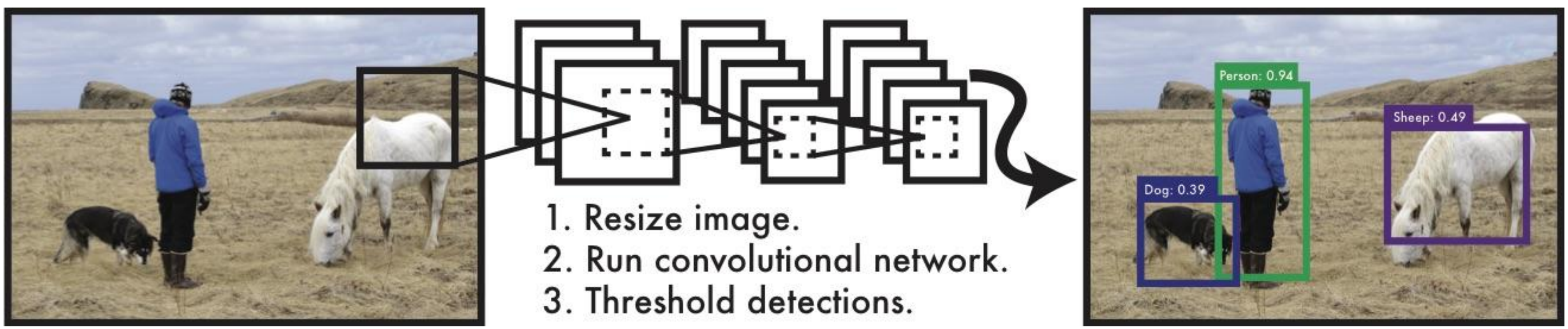
Sliding window, DPM, R-CNN all train region-based classifiers to perform detection

DPM: ***Deformable Part Models***

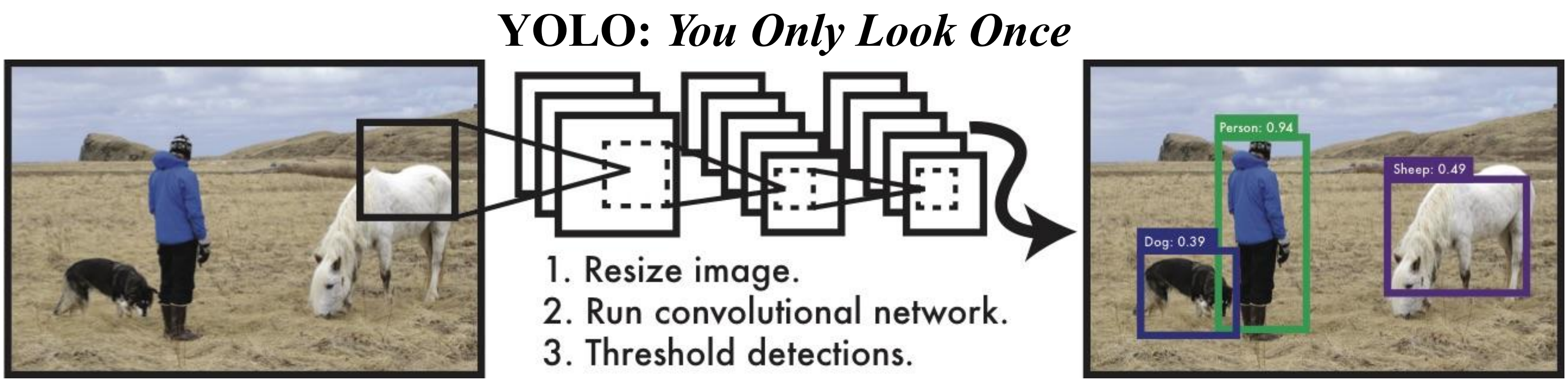


R-CNN: ***Regions with CNN features***

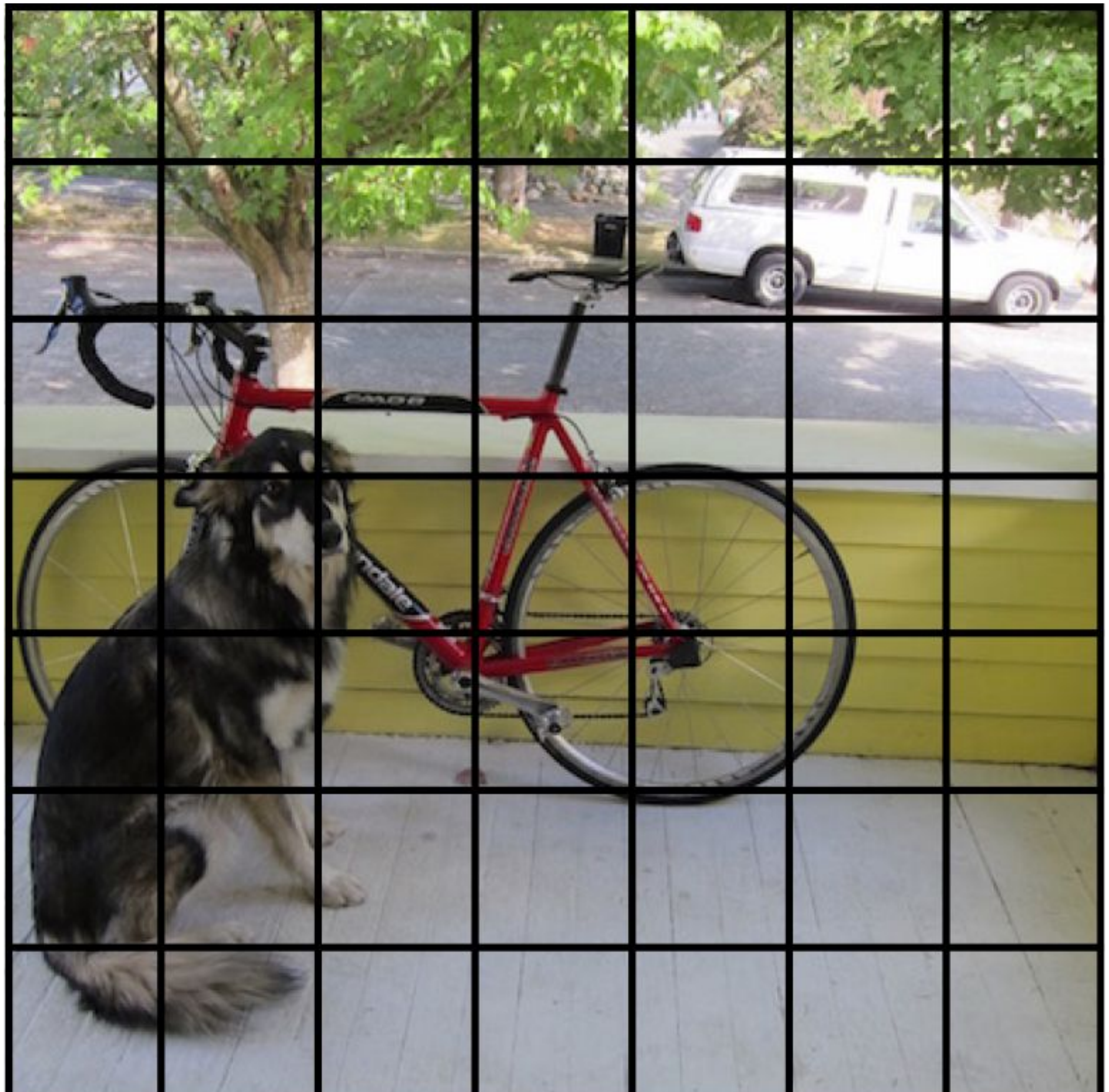



1. Resize image.
2. Run convolutional network.
3. Threshold detections.
Person: 0.94
Sheep: 0.49
Dog: 0.39

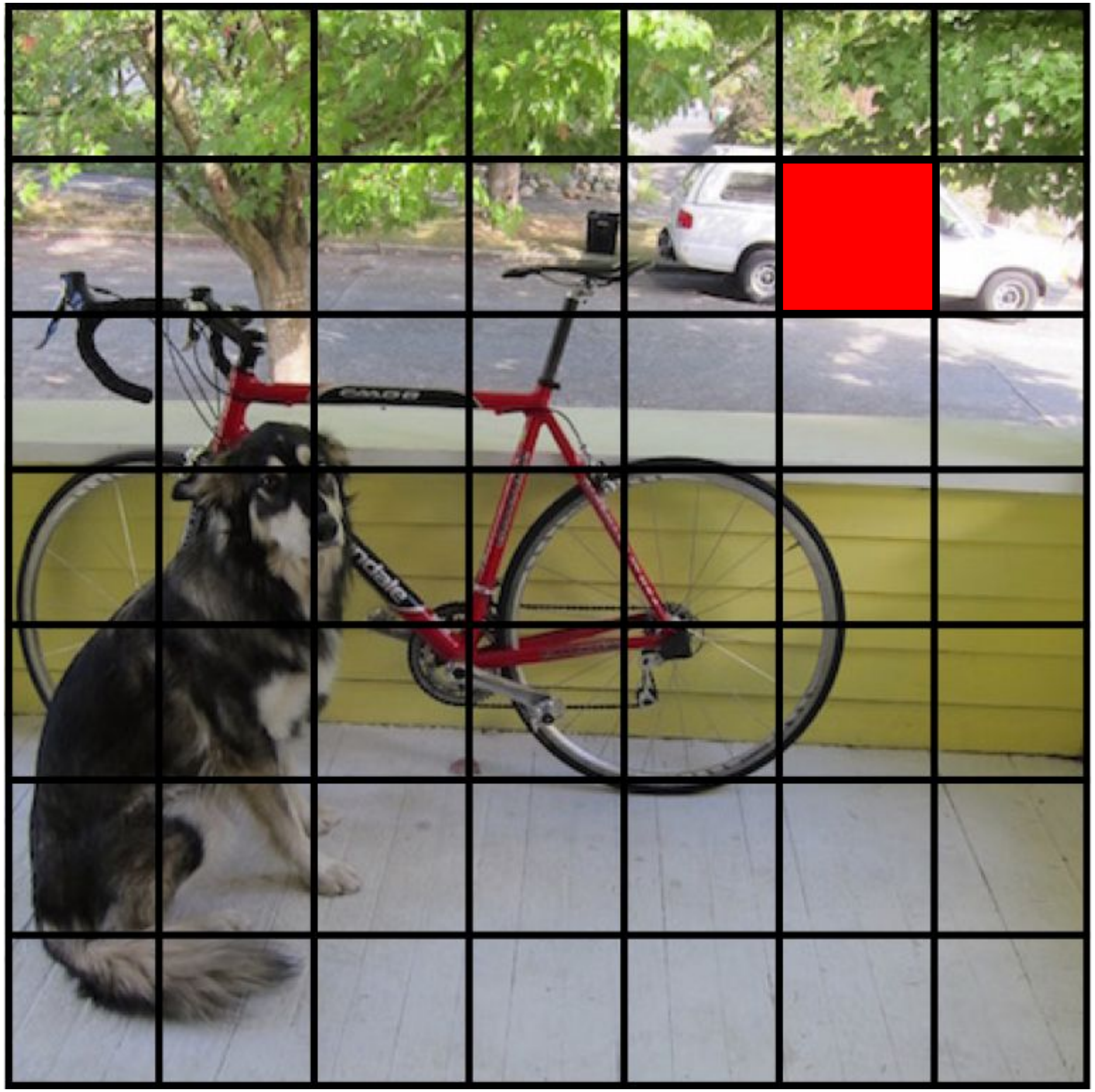
With YOLO, you only look once at an image to perform detection



We split the image into a grid

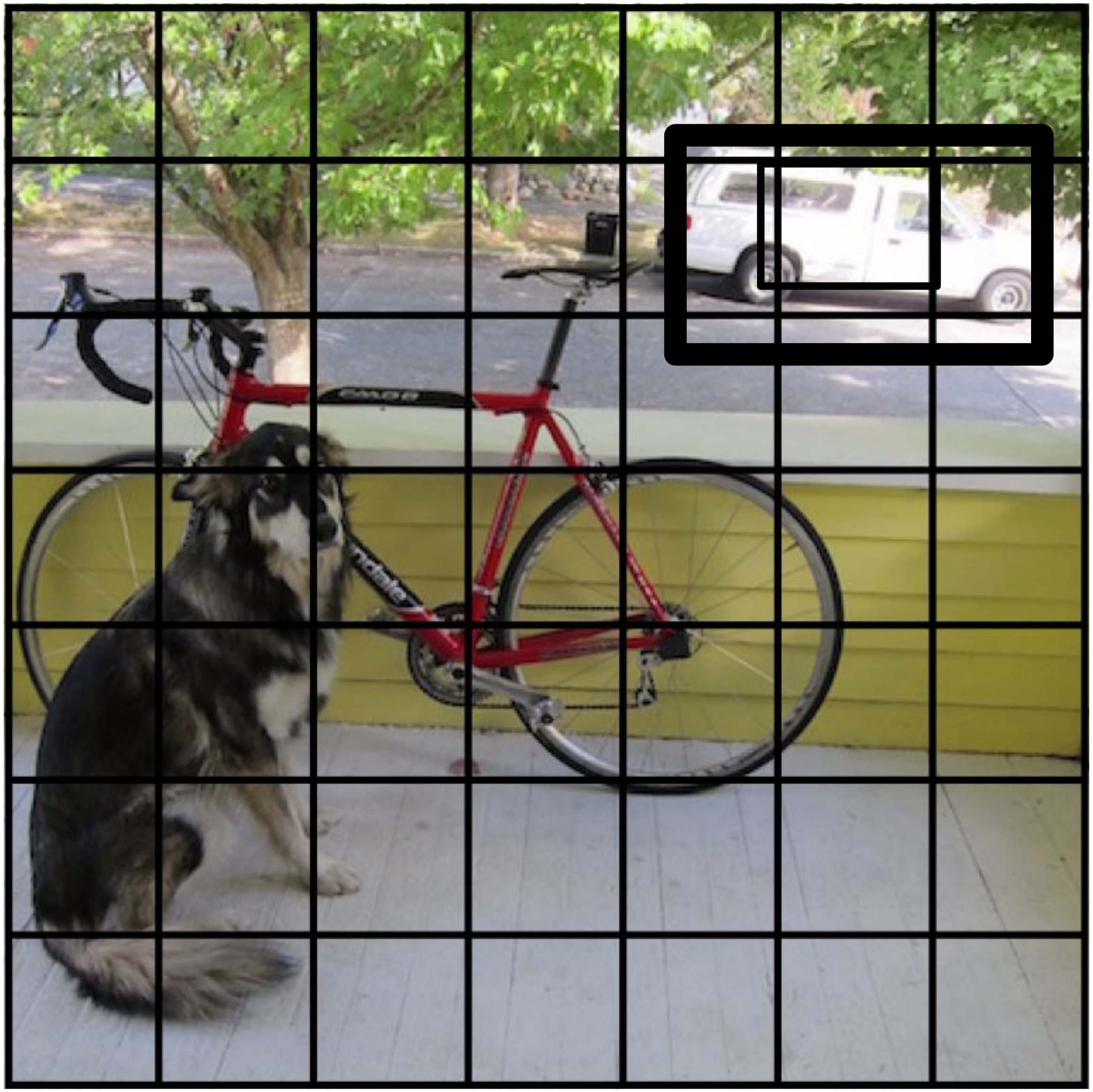


Each cell predicts boxes and confidences: P(Object)

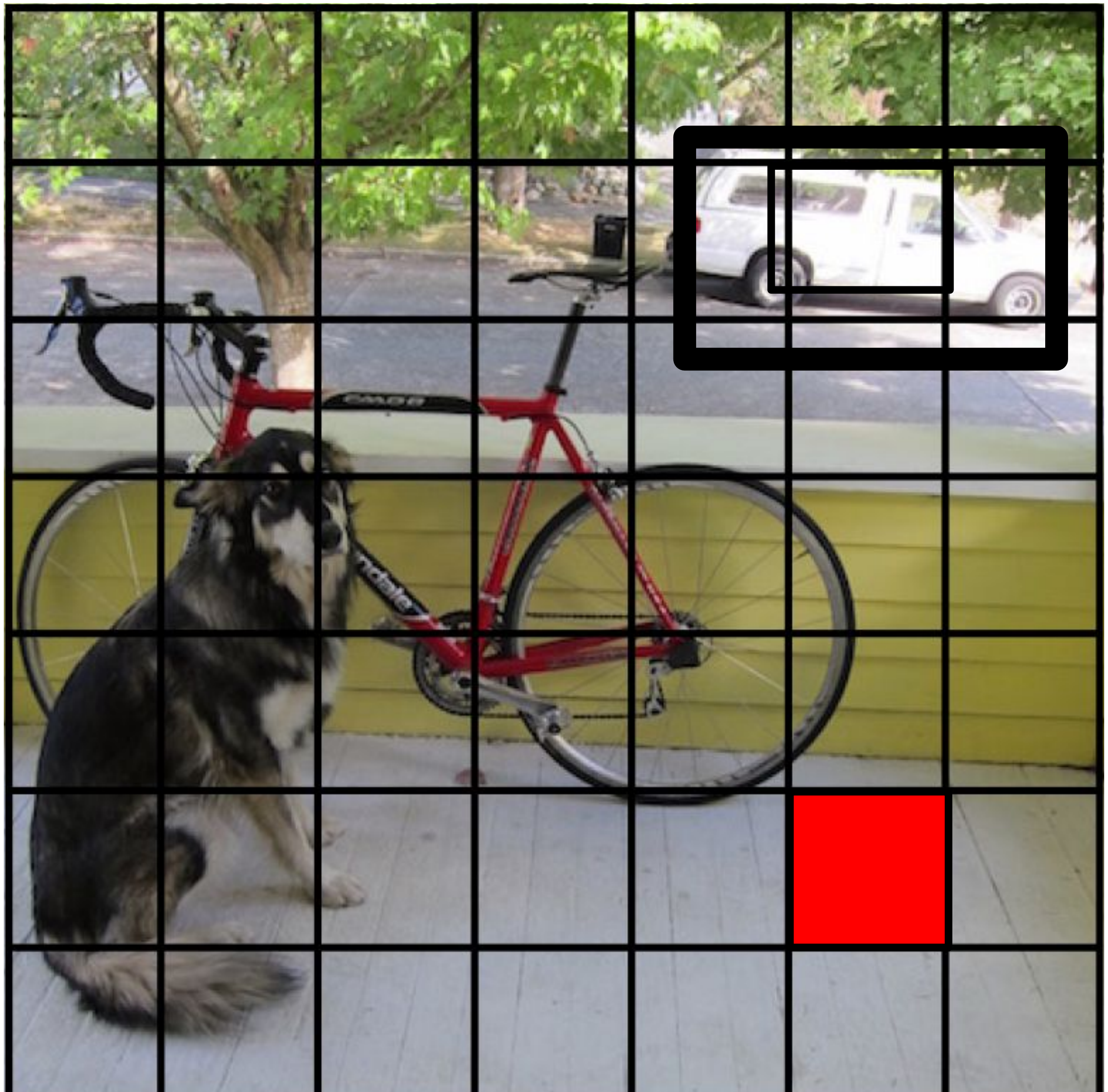


Each cell predicts boxes and confidences: P(Object)

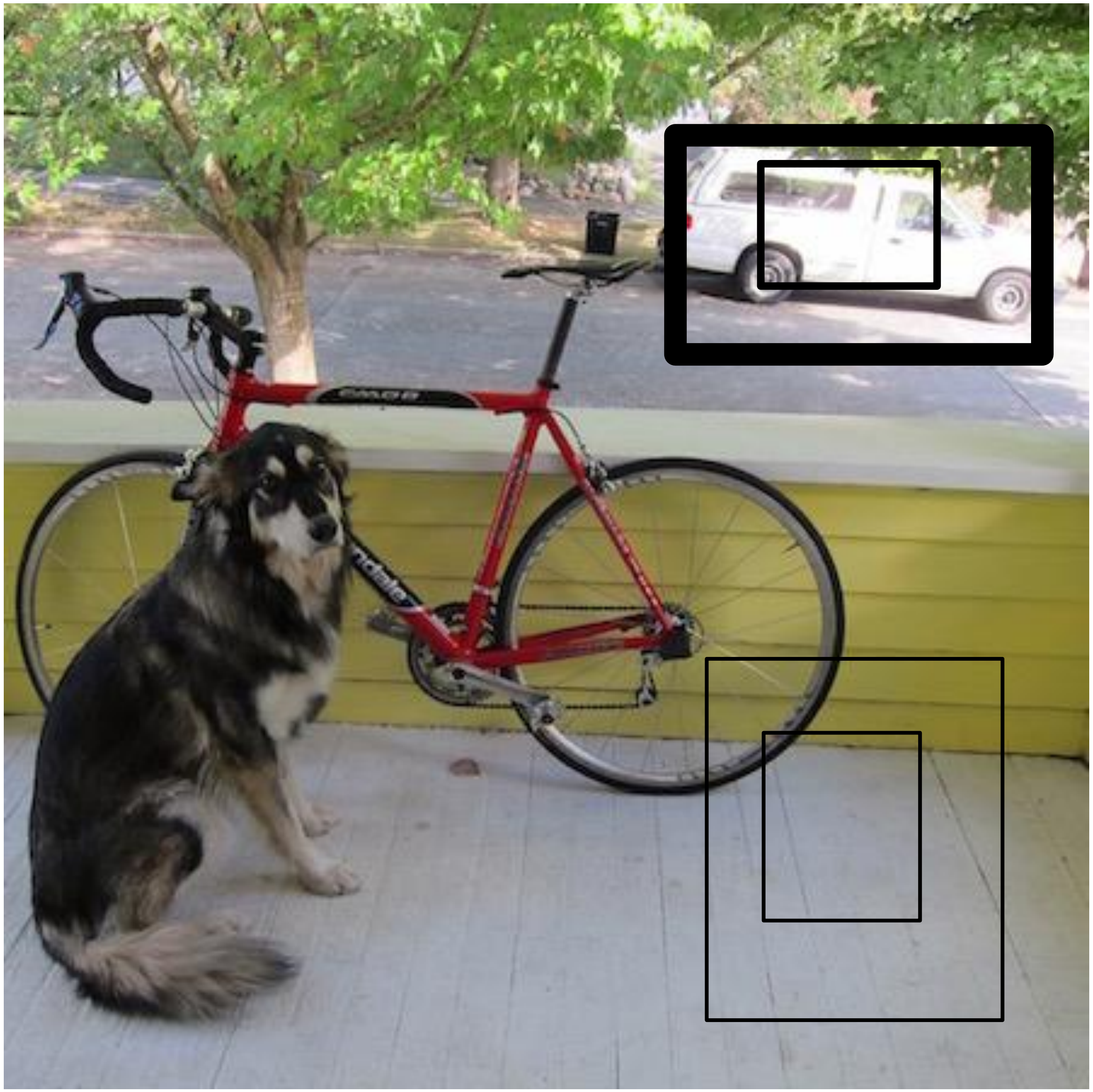
Each cell predicts boxes and confidences: P(Object)



Each cell predicts boxes and confidences: P(Object)

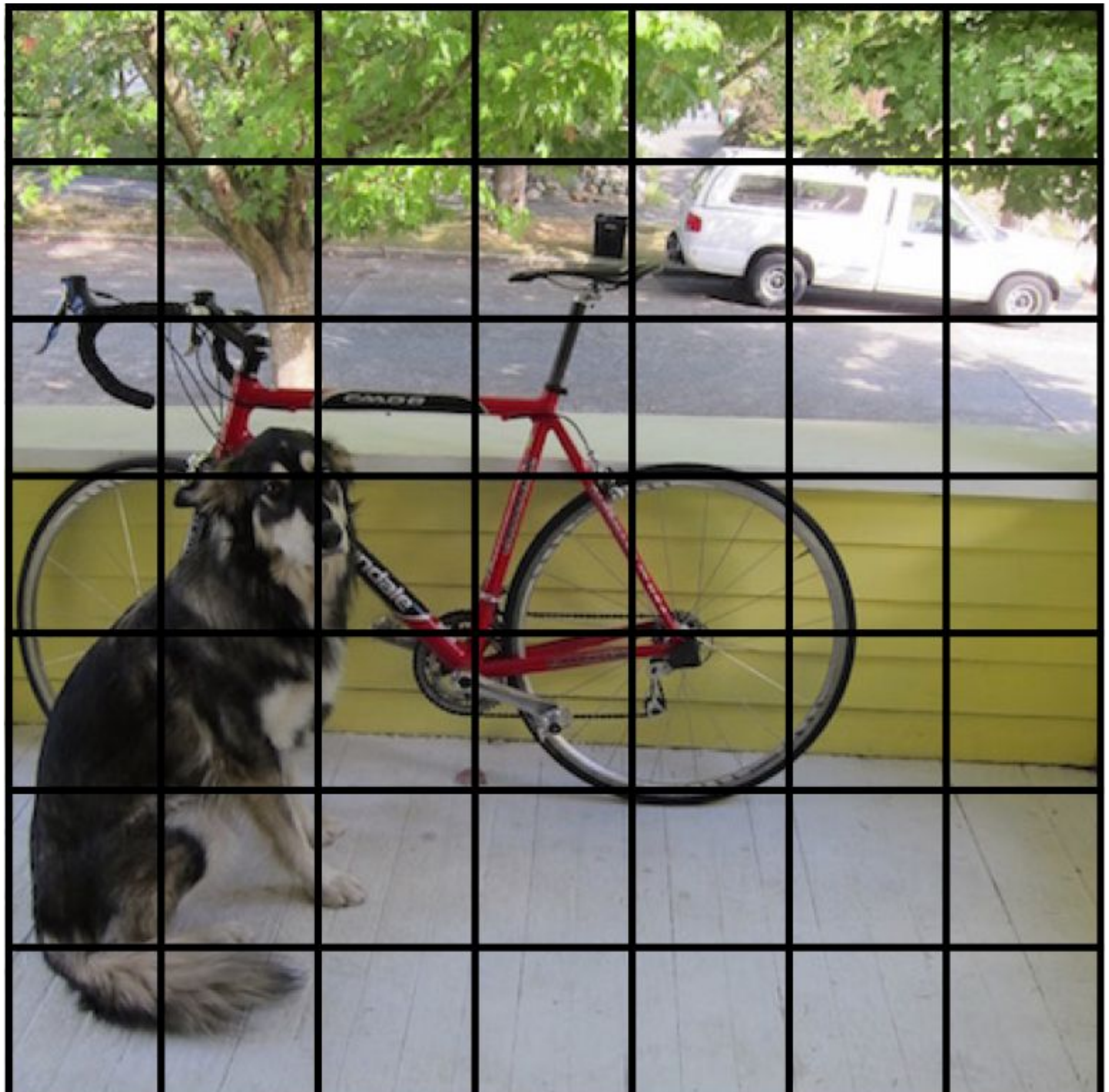


Each cell predicts boxes and confidences: P(Object)

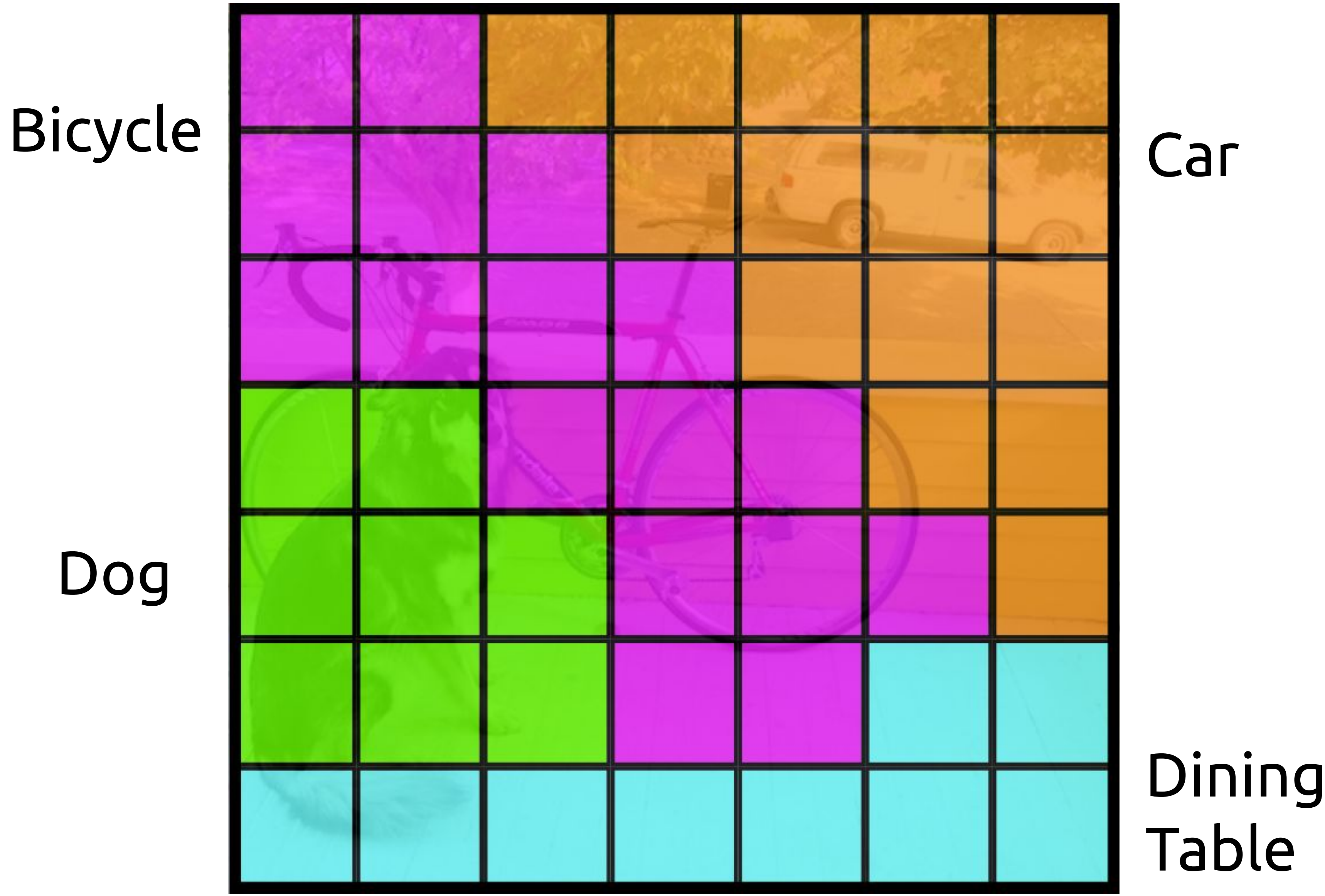


Each cell predicts boxes and confidences: P(Object)

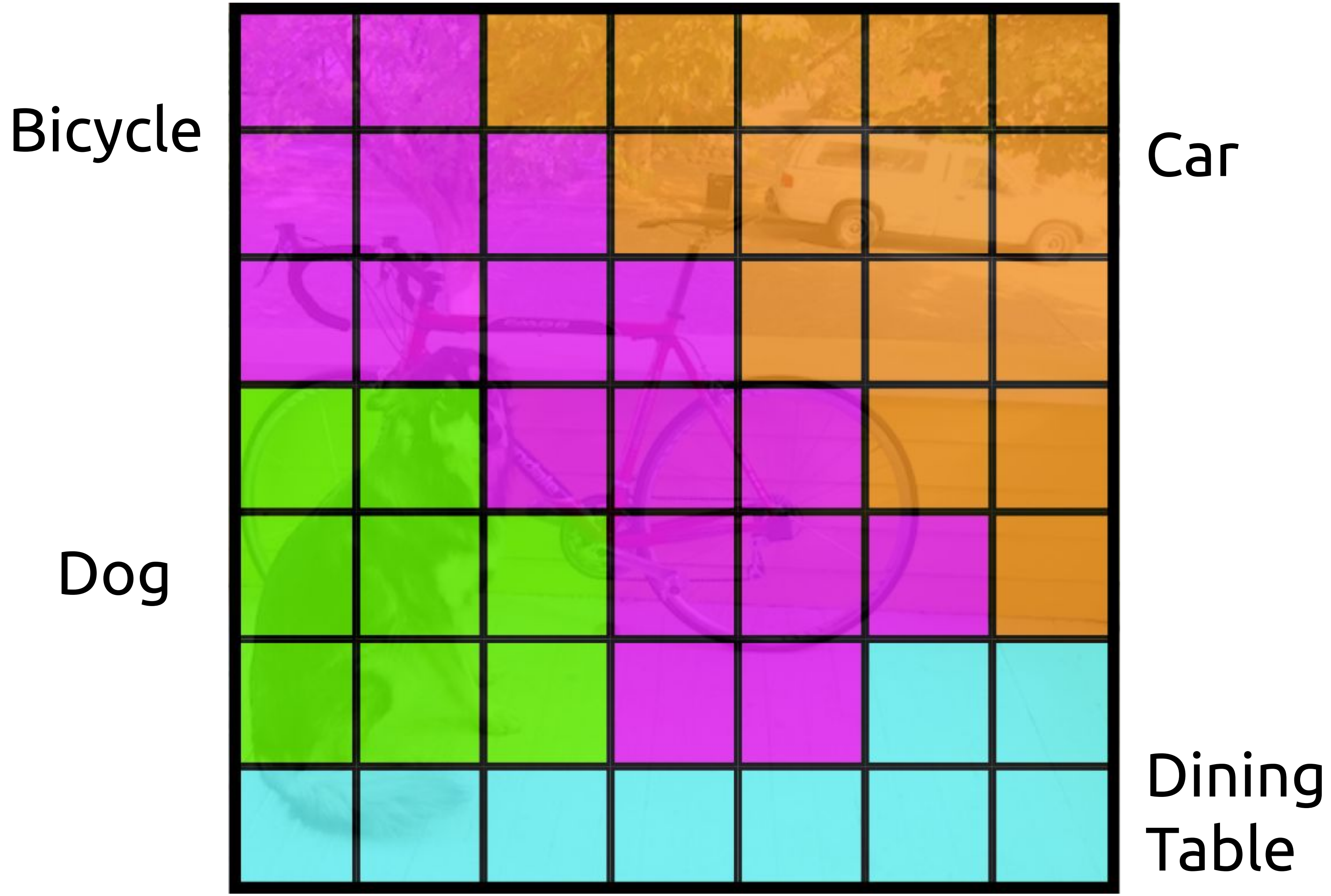
Each cell also predicts a class probability.



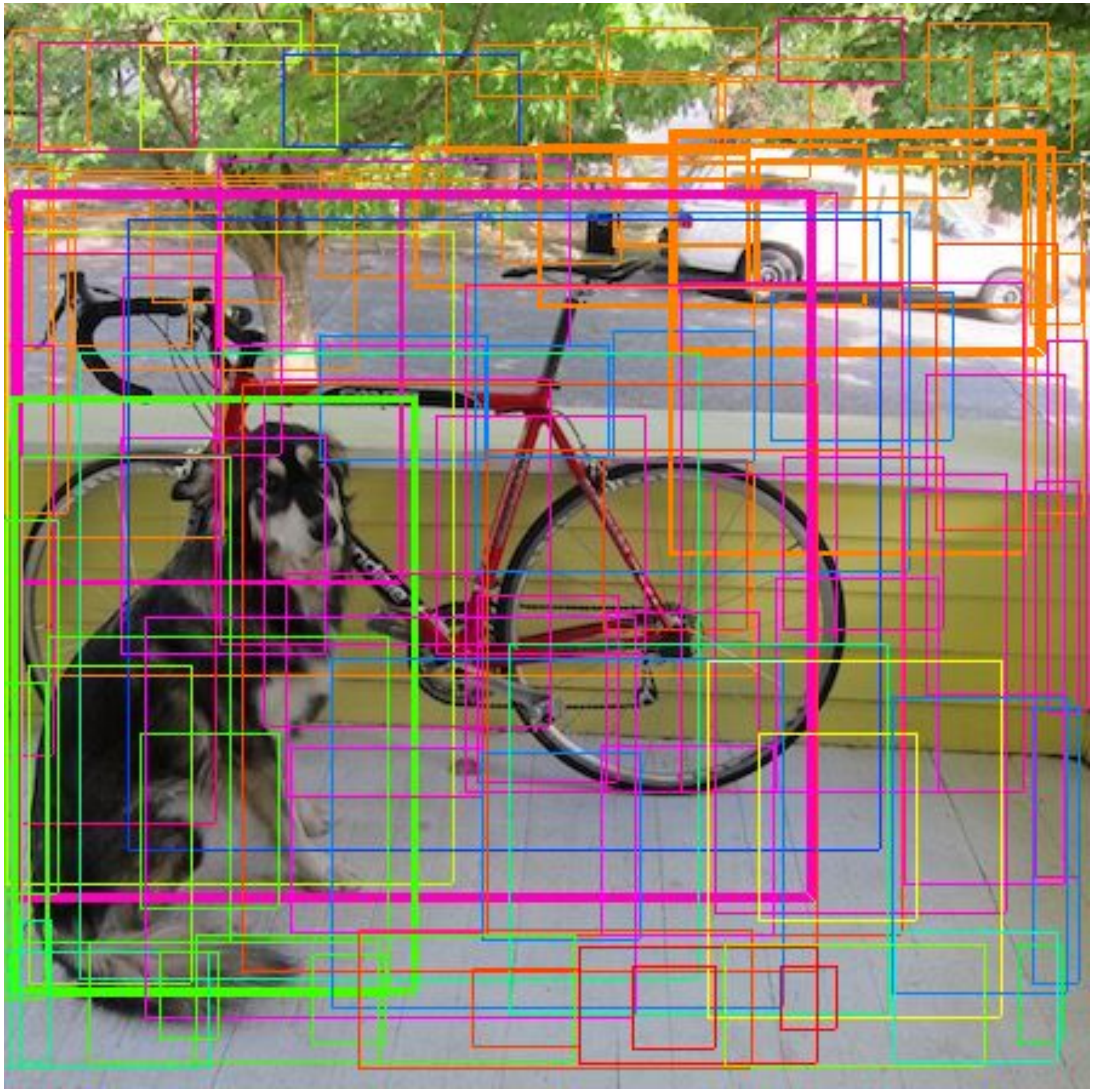
Each cell also predicts a class probability.



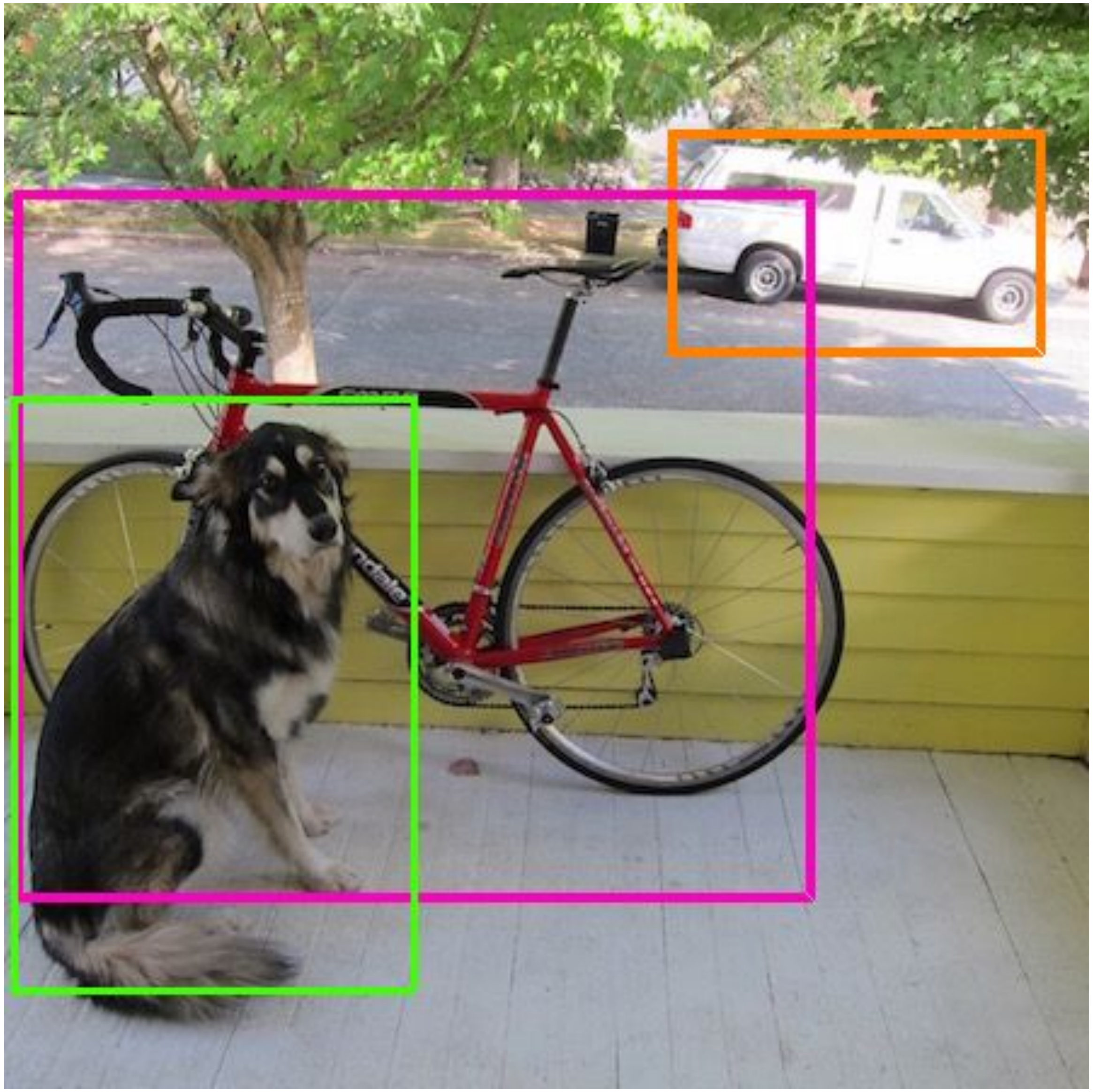
Conditioned on object: P(Car | Object)



Then we combine the box and class predictions.



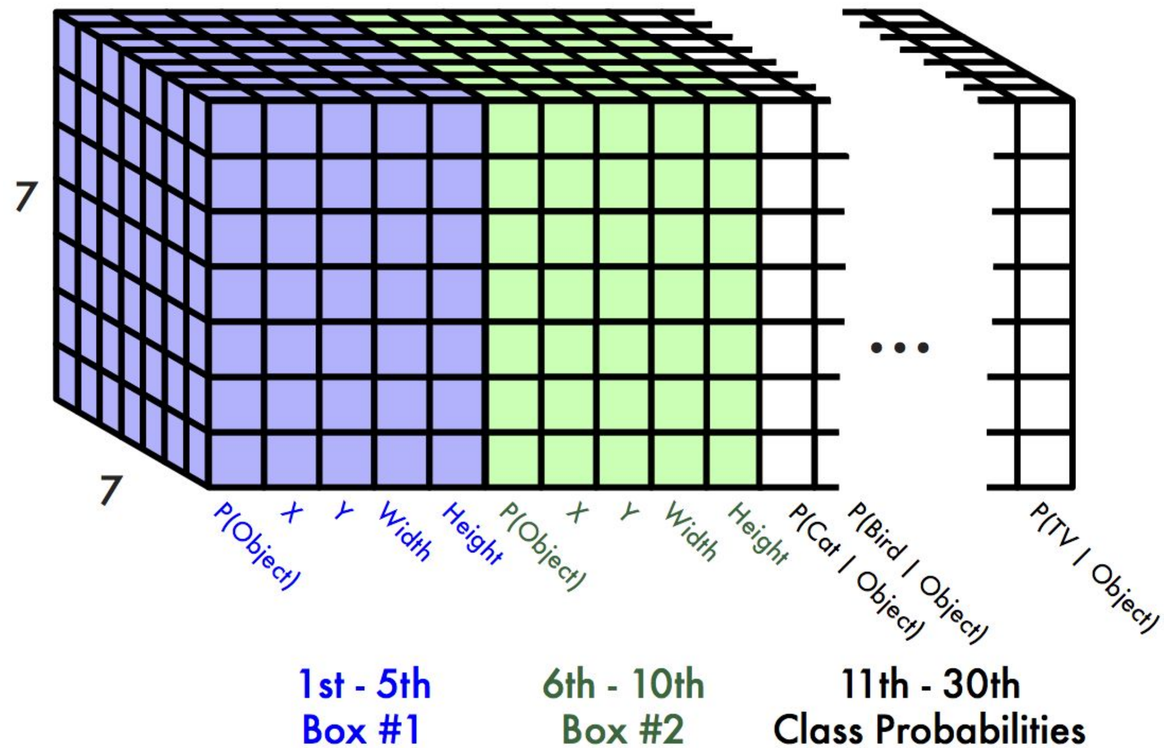
Finally we do NMS and threshold detections



This parameterization fixes the output size

Each cell predicts:

- For each bounding box:
  - 4 coordinates (x, y, w, h)
  - 1 confidence value
- Some number of class probabilities

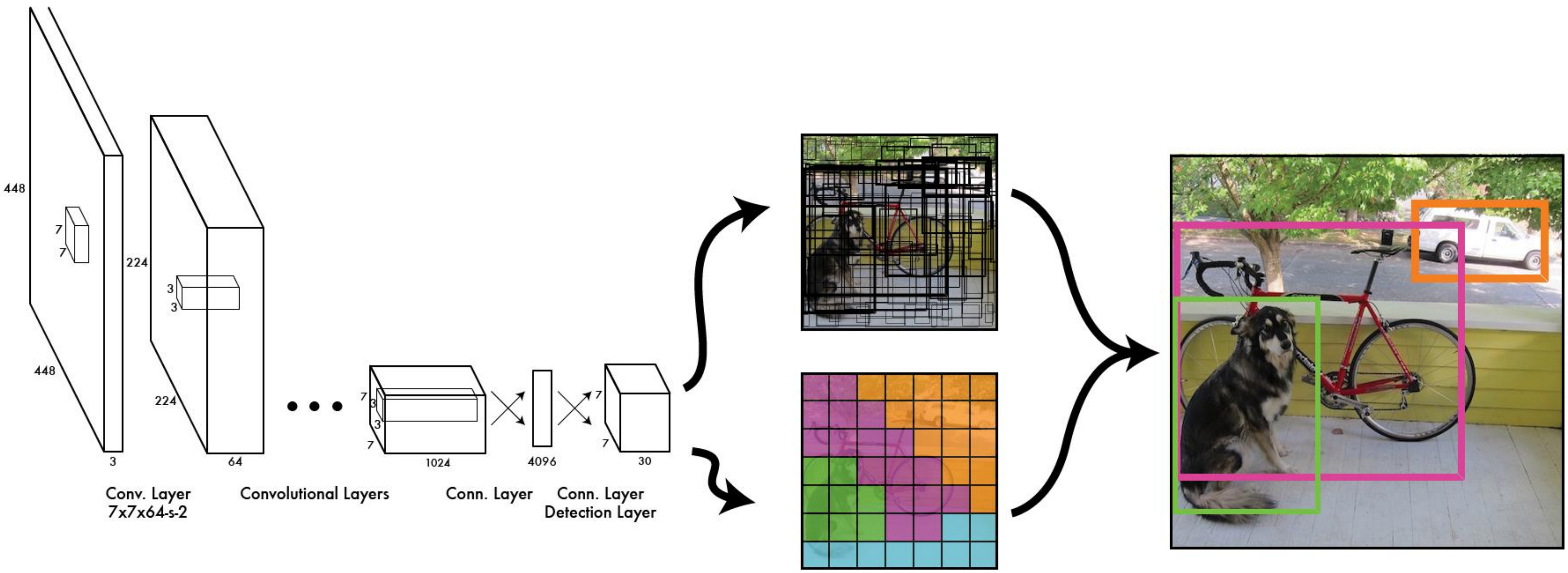


For Pascal VOC:

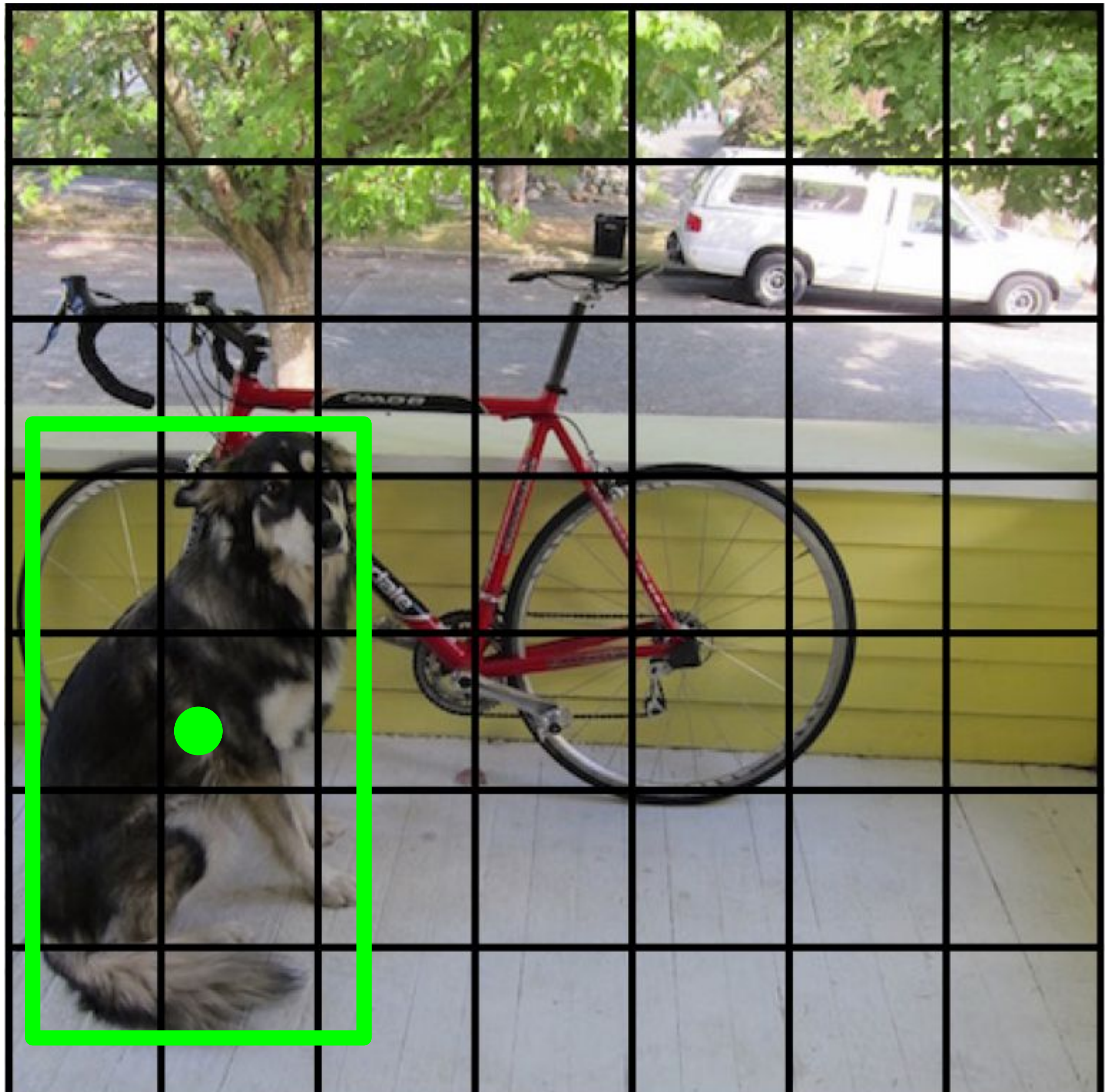
- 7x7 grid
- 2 bounding boxes / cell
- 20 classes

7 x 7 x (2 x 5 + 20) = 7 x 7 x 30 tensor = **1470 outputs**

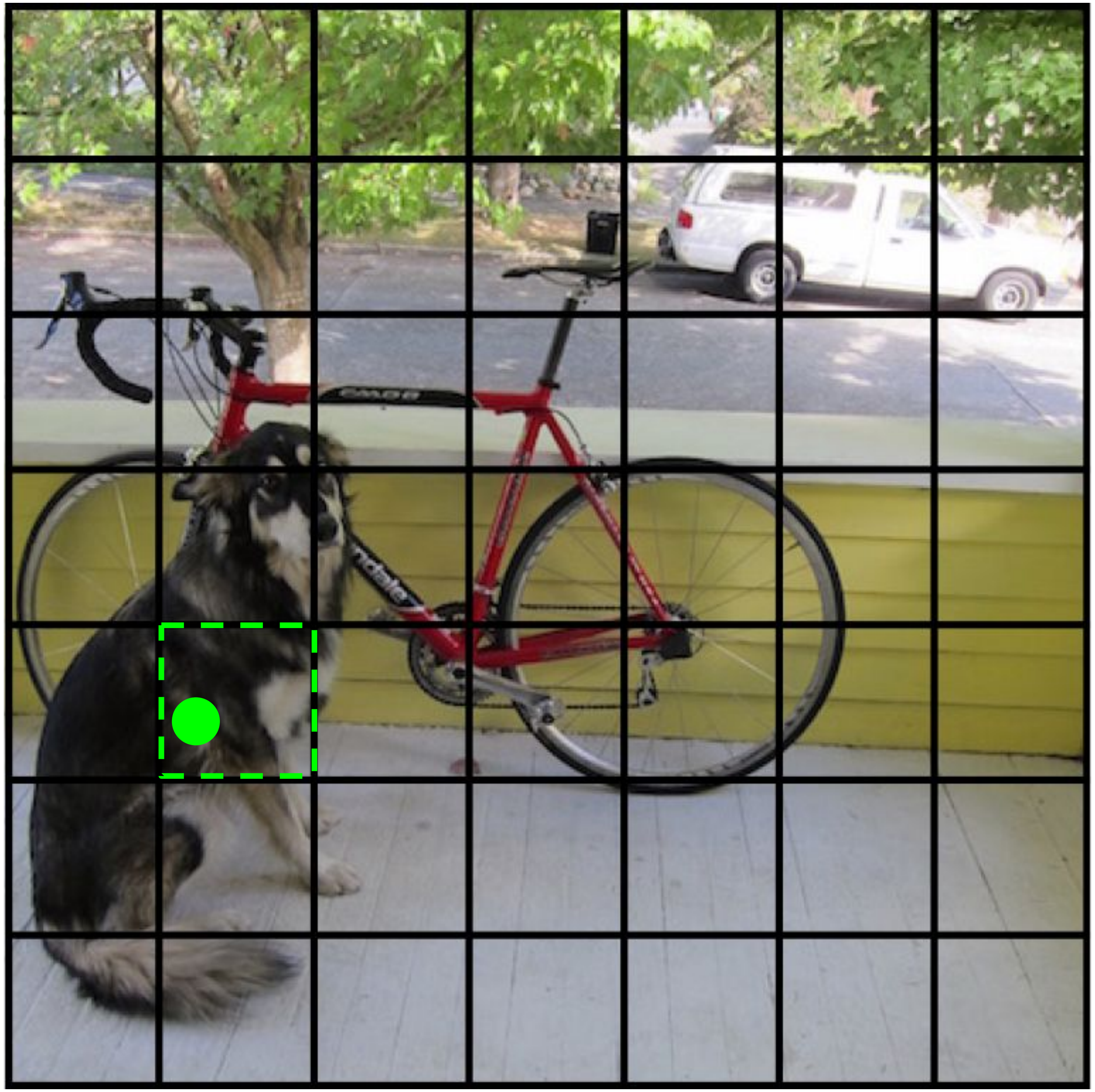
Thus we can train one neural network to be a whole detection pipeline



During training, match example to the right cell

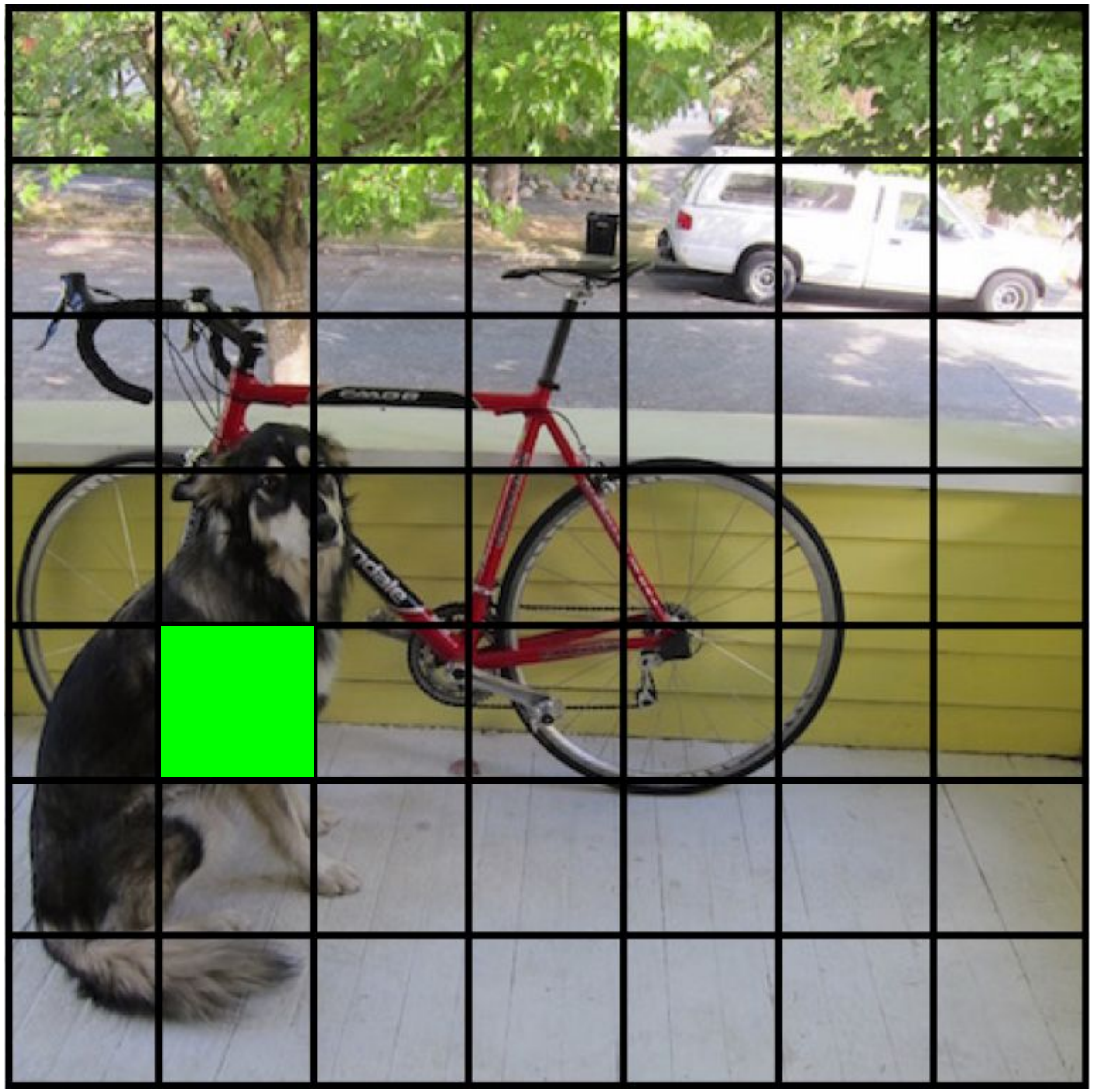


During training, match example to the right cell

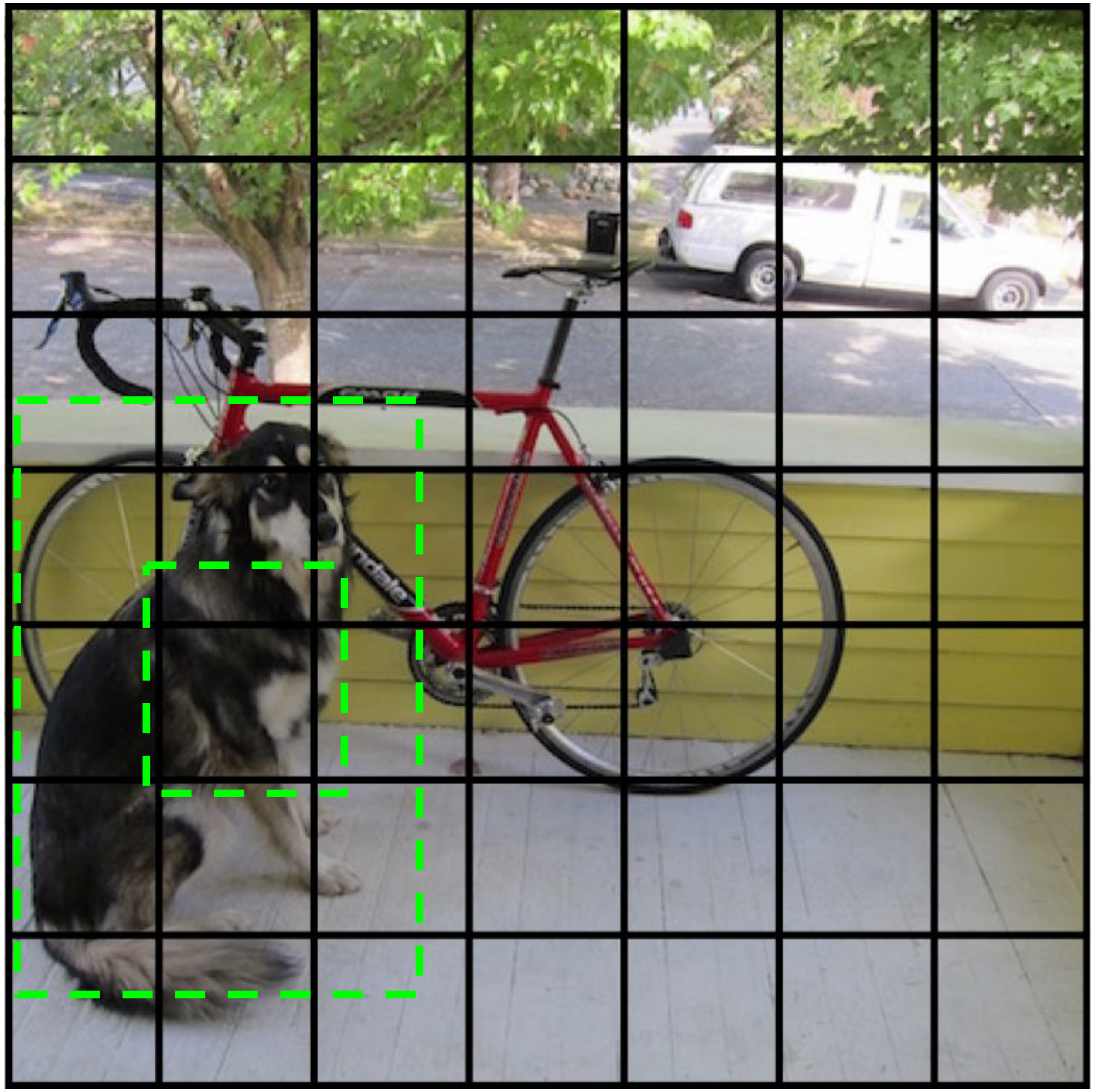


Adjust that cell's class prediction

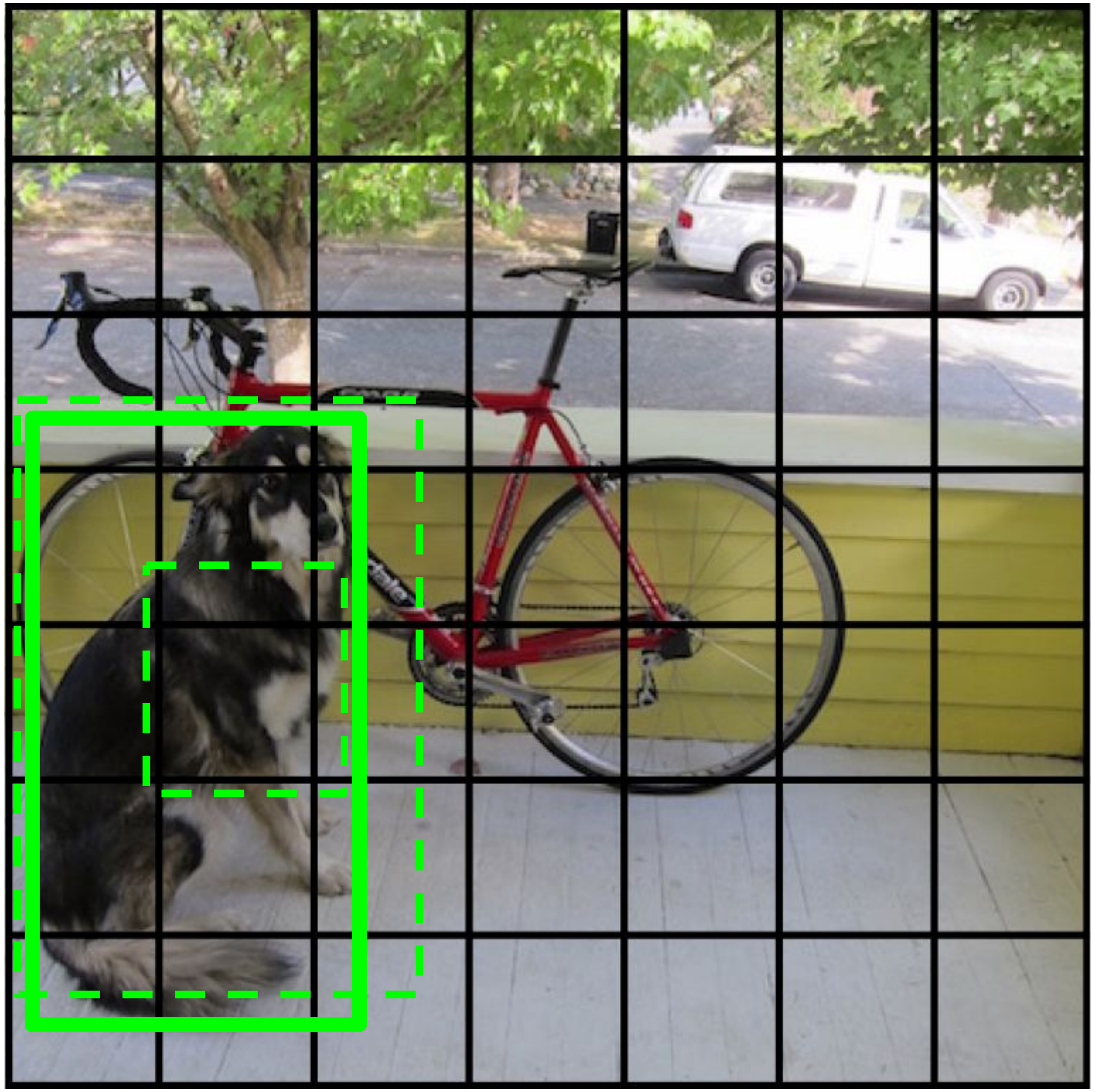
Dog = 1
Cat = 0
Bike = 0
...



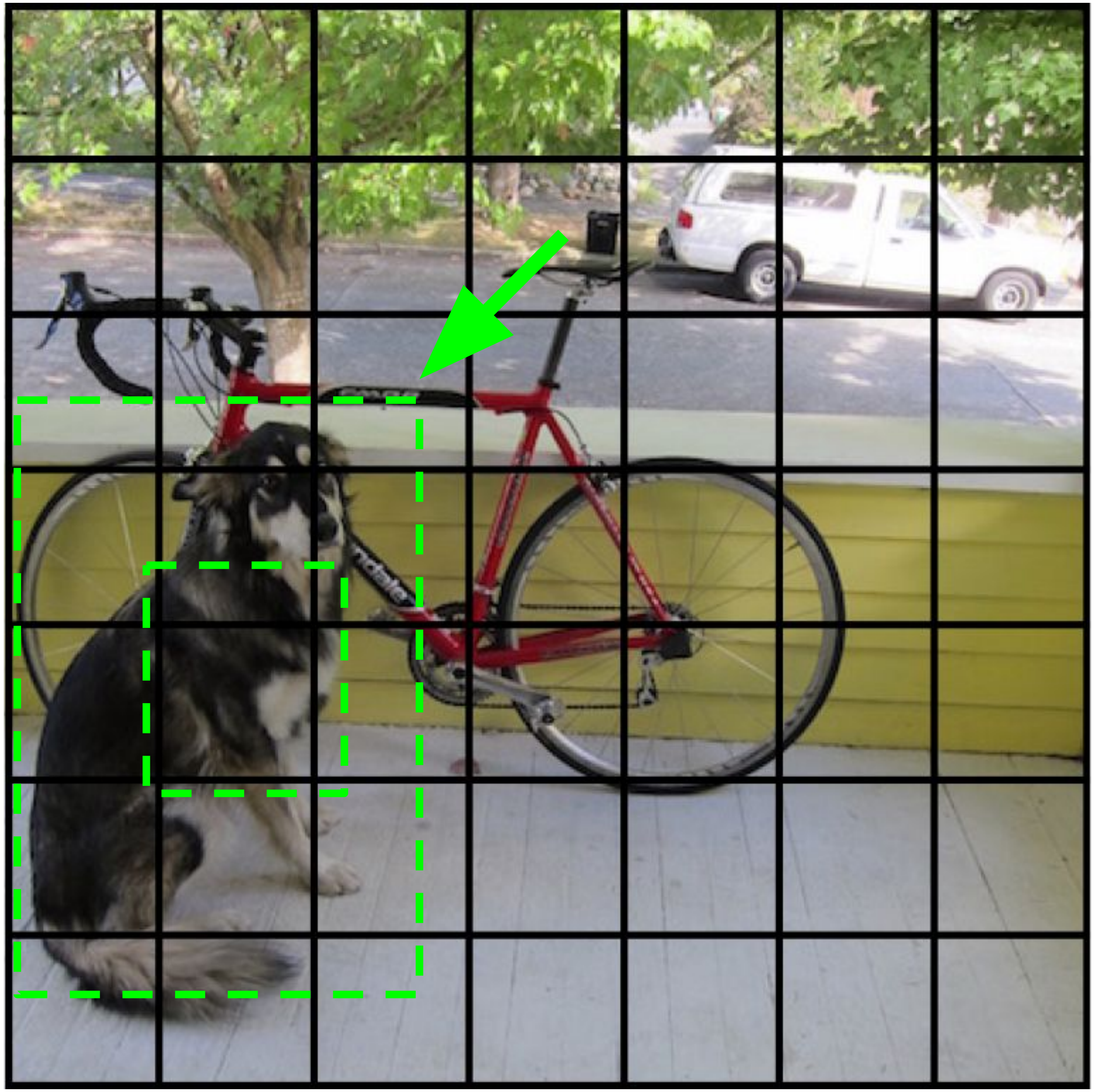
Look at that cell's predicted boxes



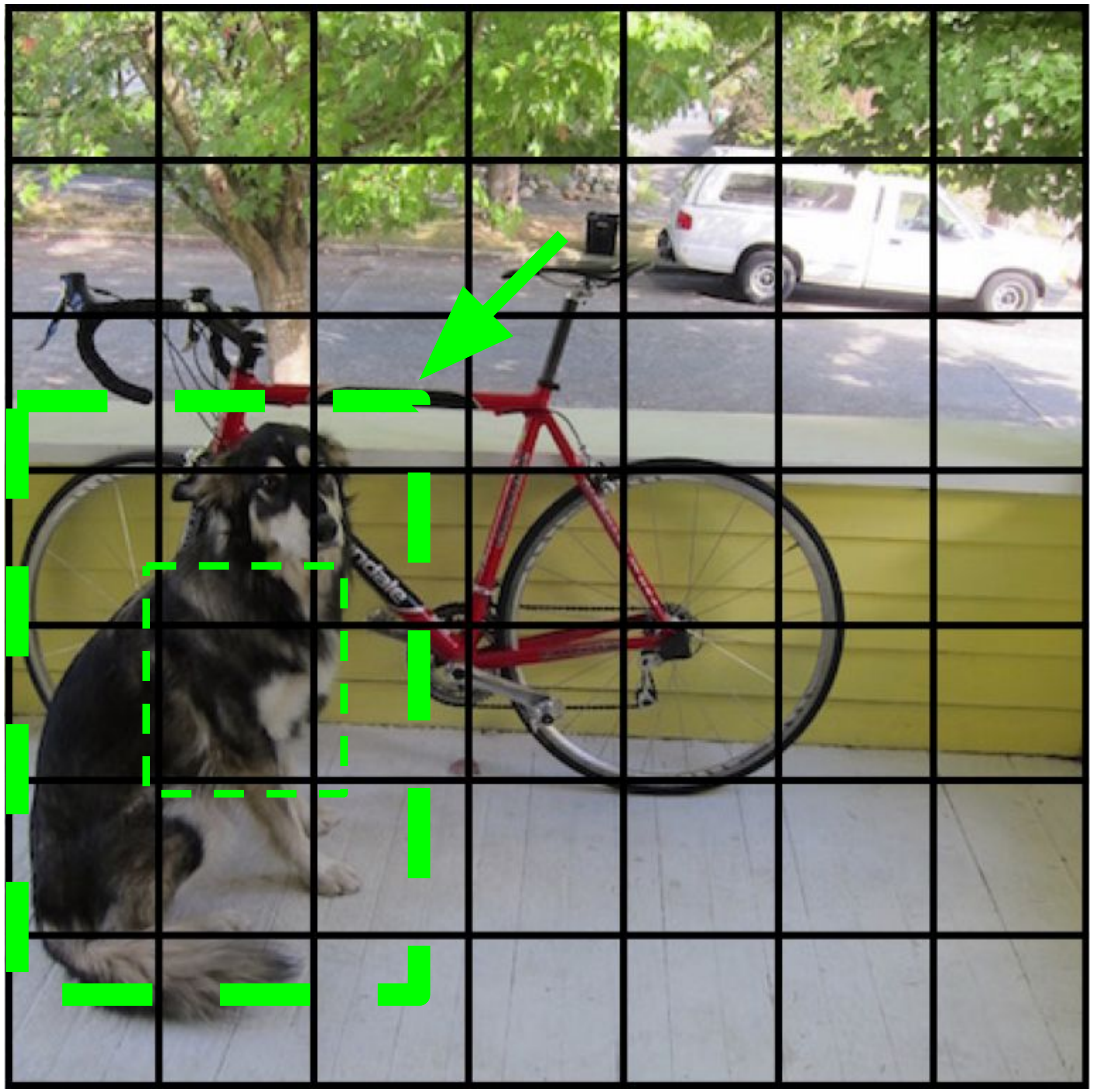
Find the best one, adjust it, increase the confidence



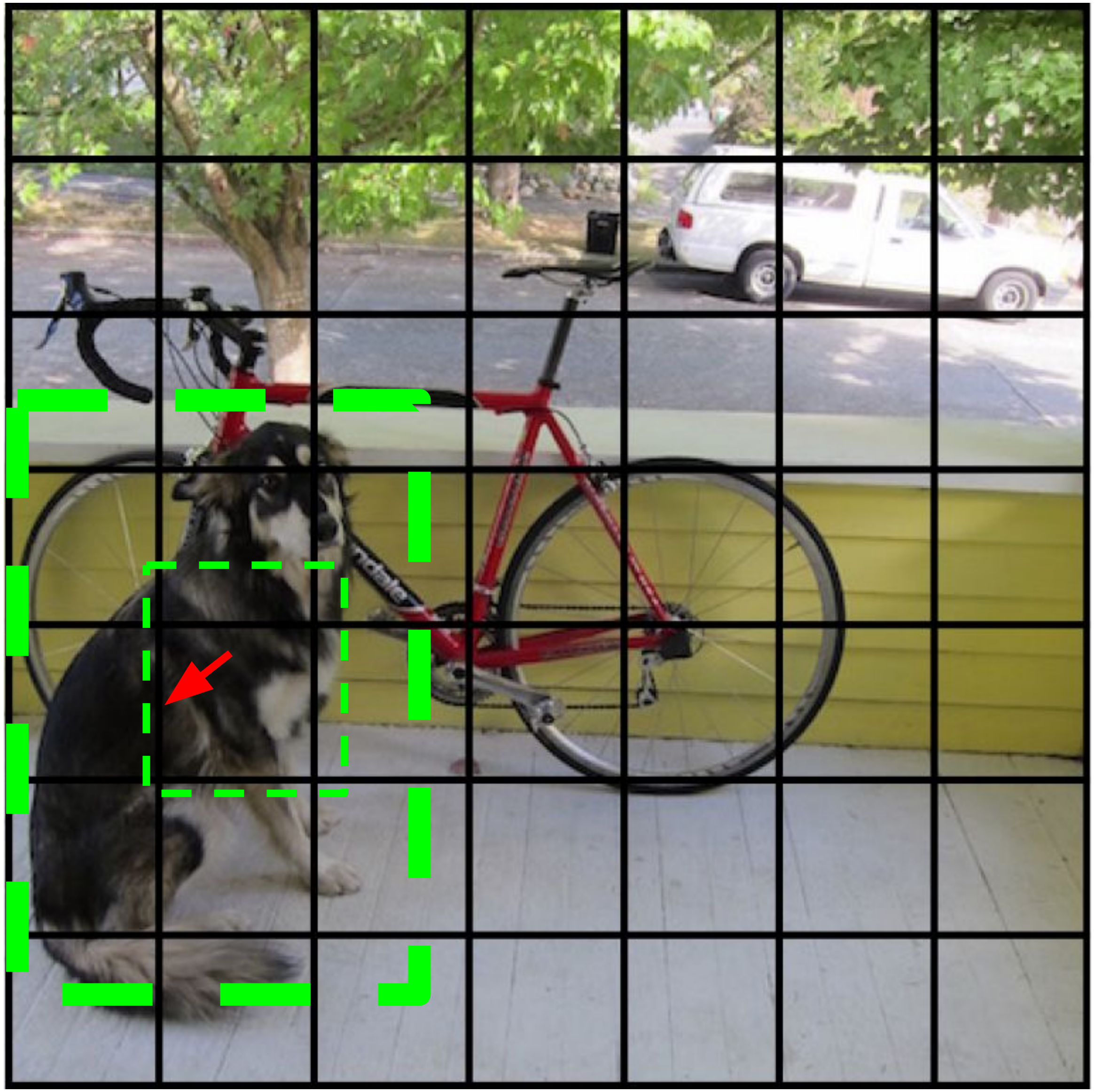
Find the best one, adjust it, increase the confidence



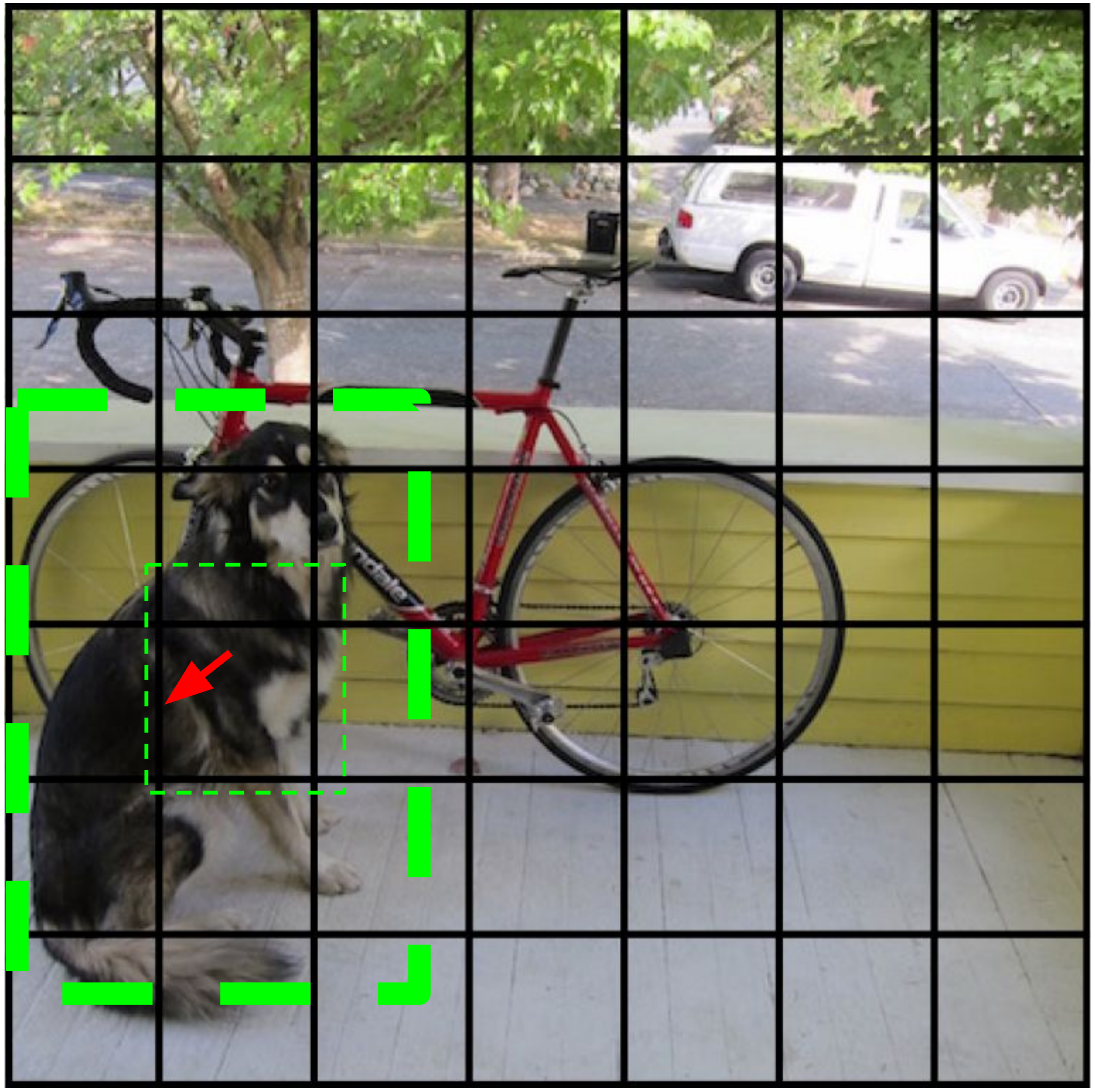
Find the best one, adjust it, increase the confidence



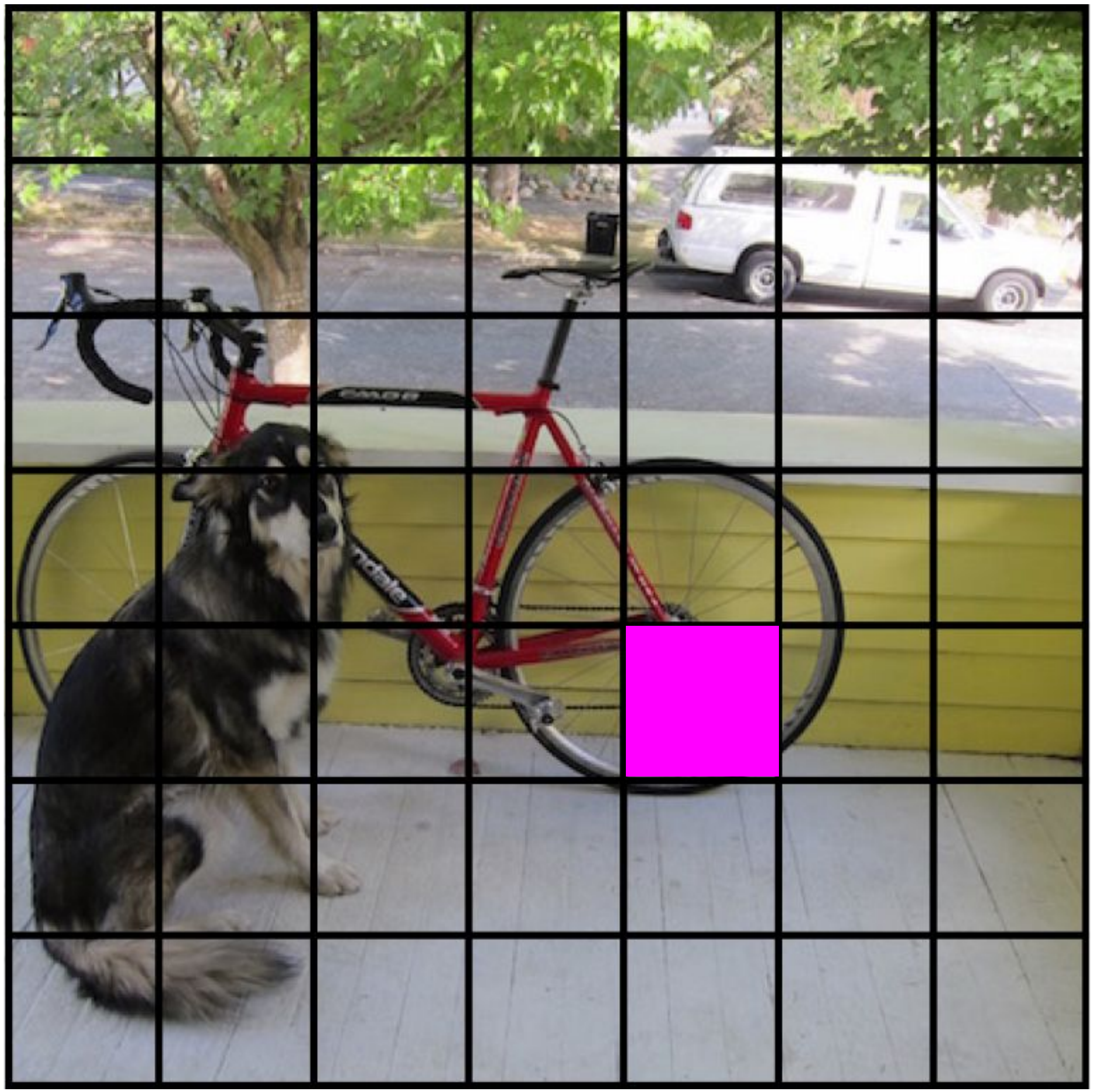
Decrease the confidence of other boxes



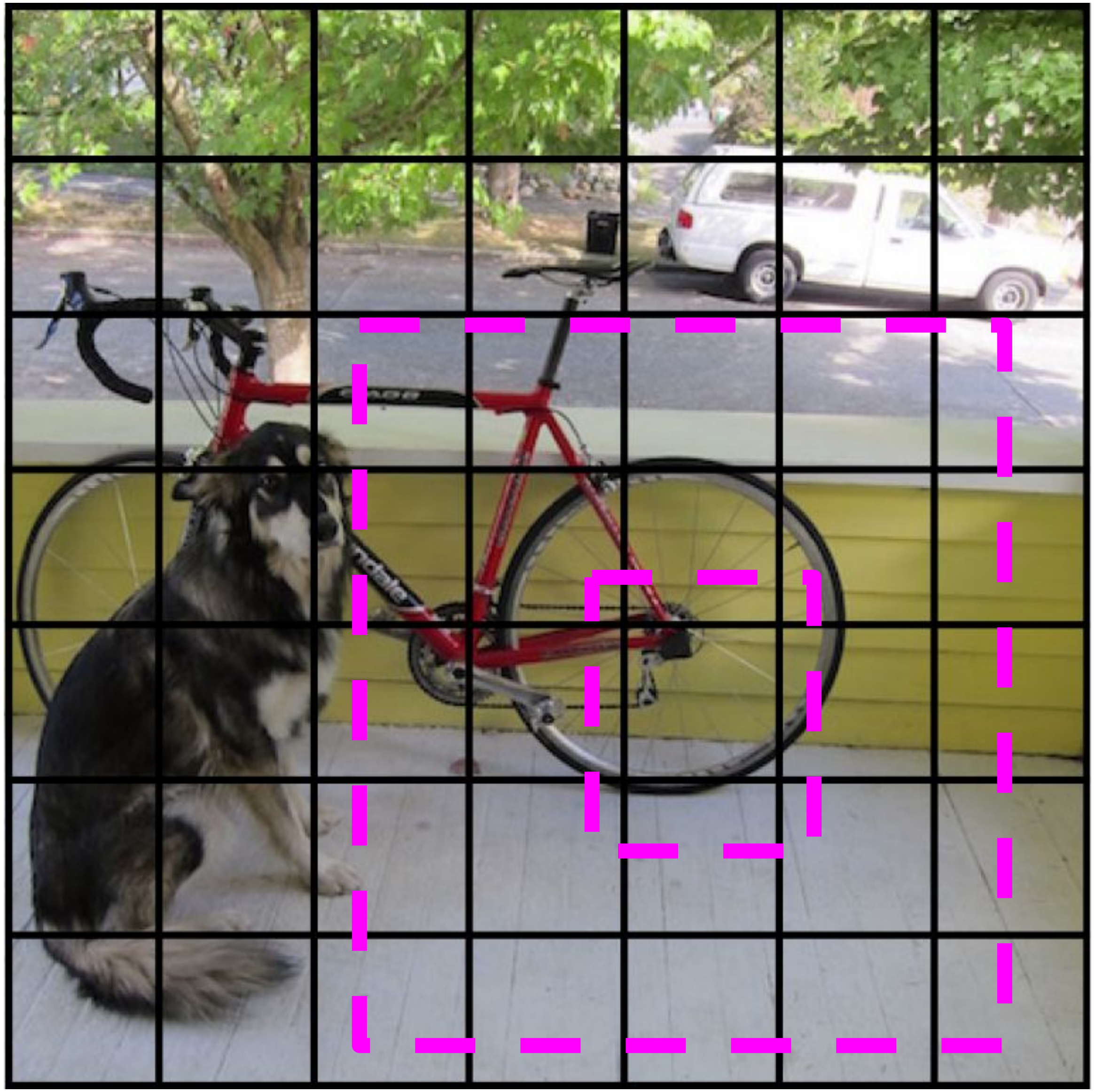
Decrease the confidence of other boxes



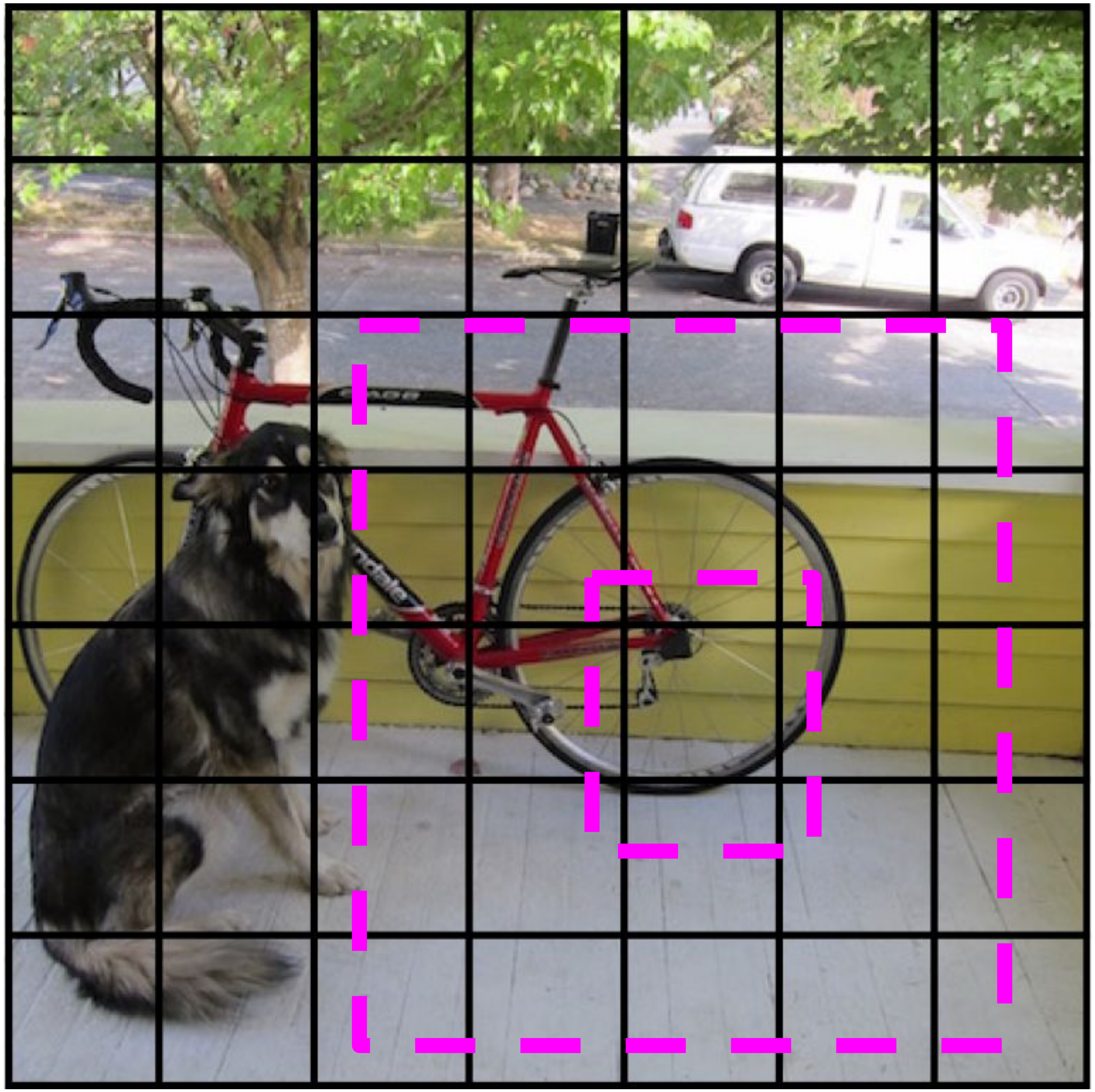
Some cells don't have any ground truth detections!



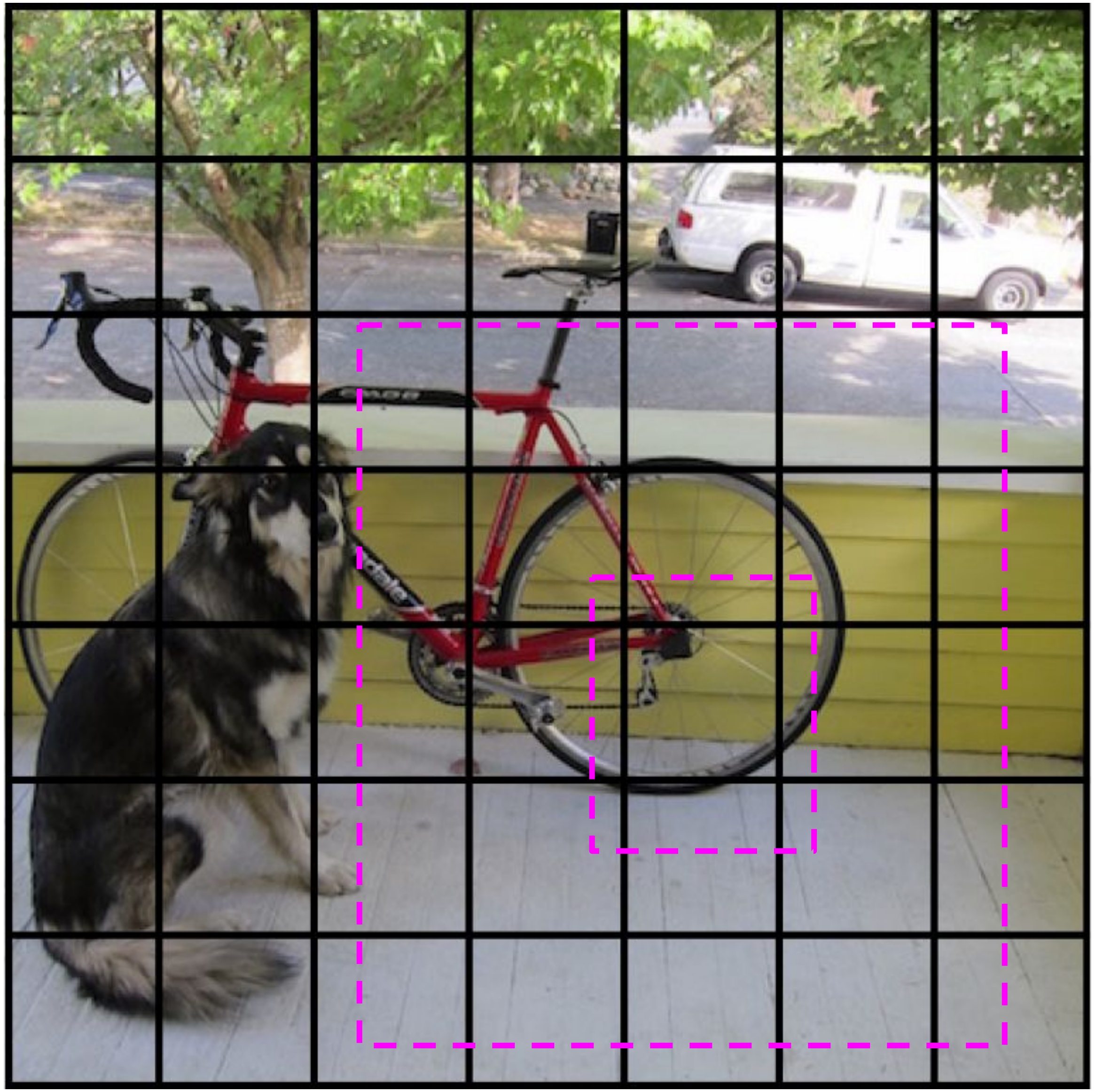
Some cells don’t have any ground truth detections!



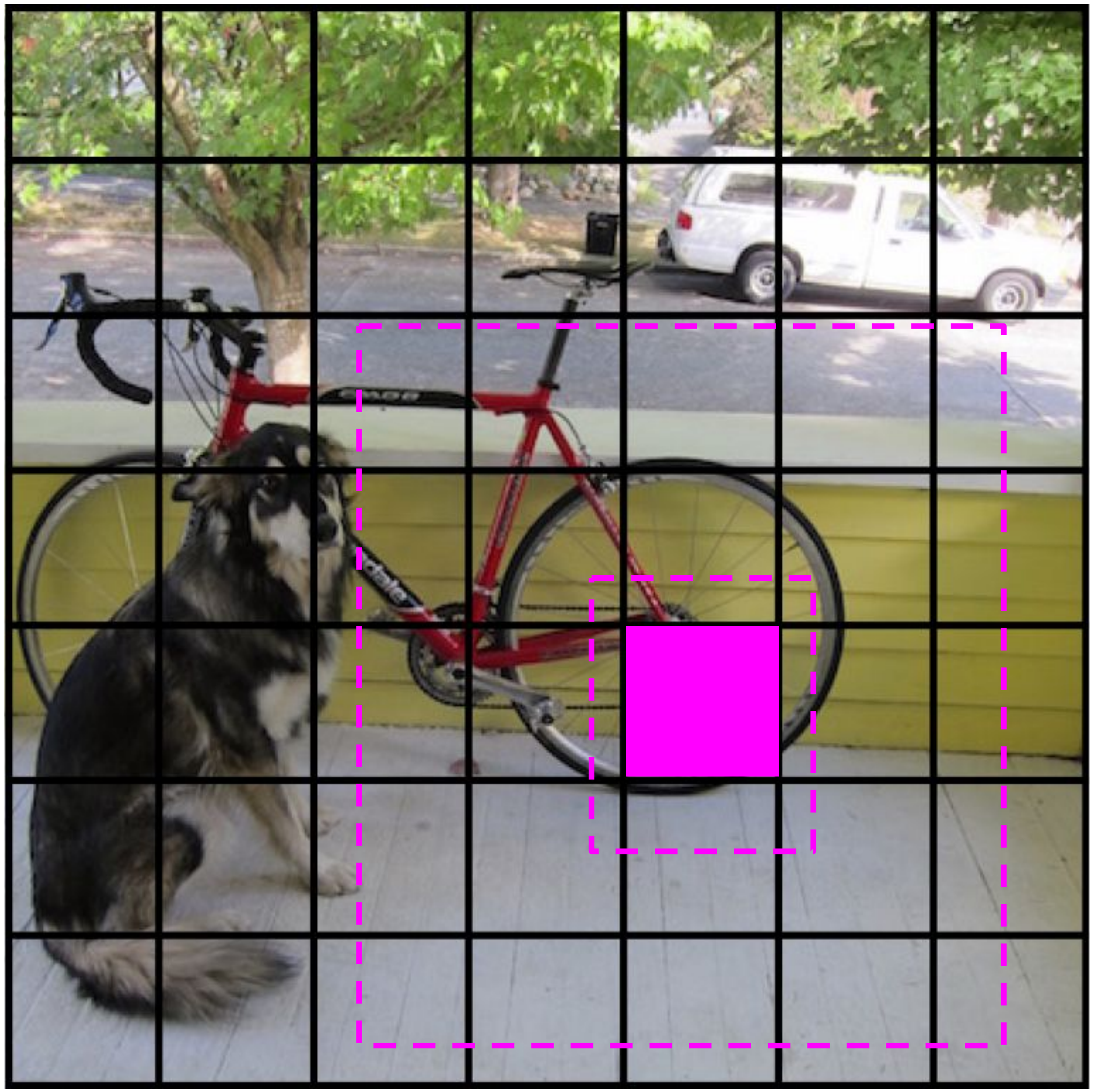
Decrease the confidence of these boxes



Decrease the confidence of these boxes

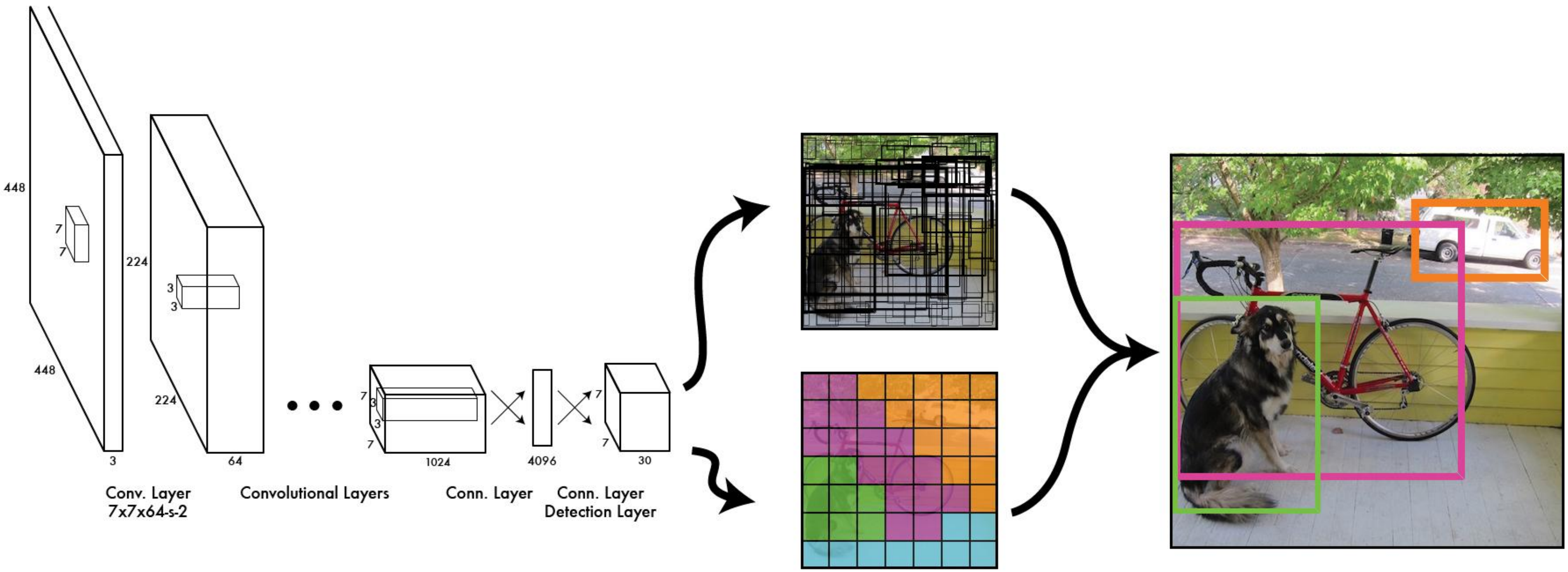


Don’t adjust the class probabilities or coordinates

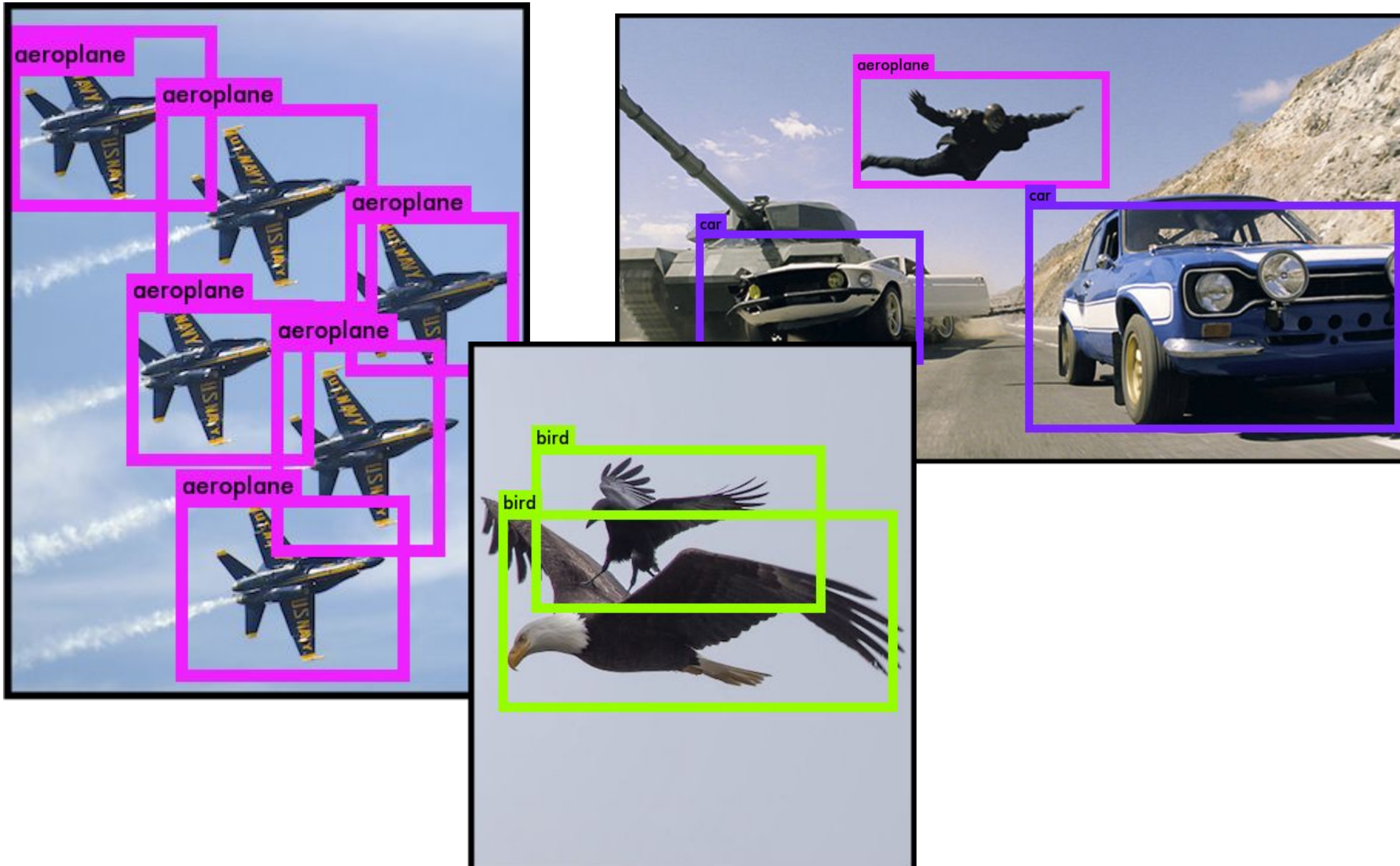


We train with standard tricks:

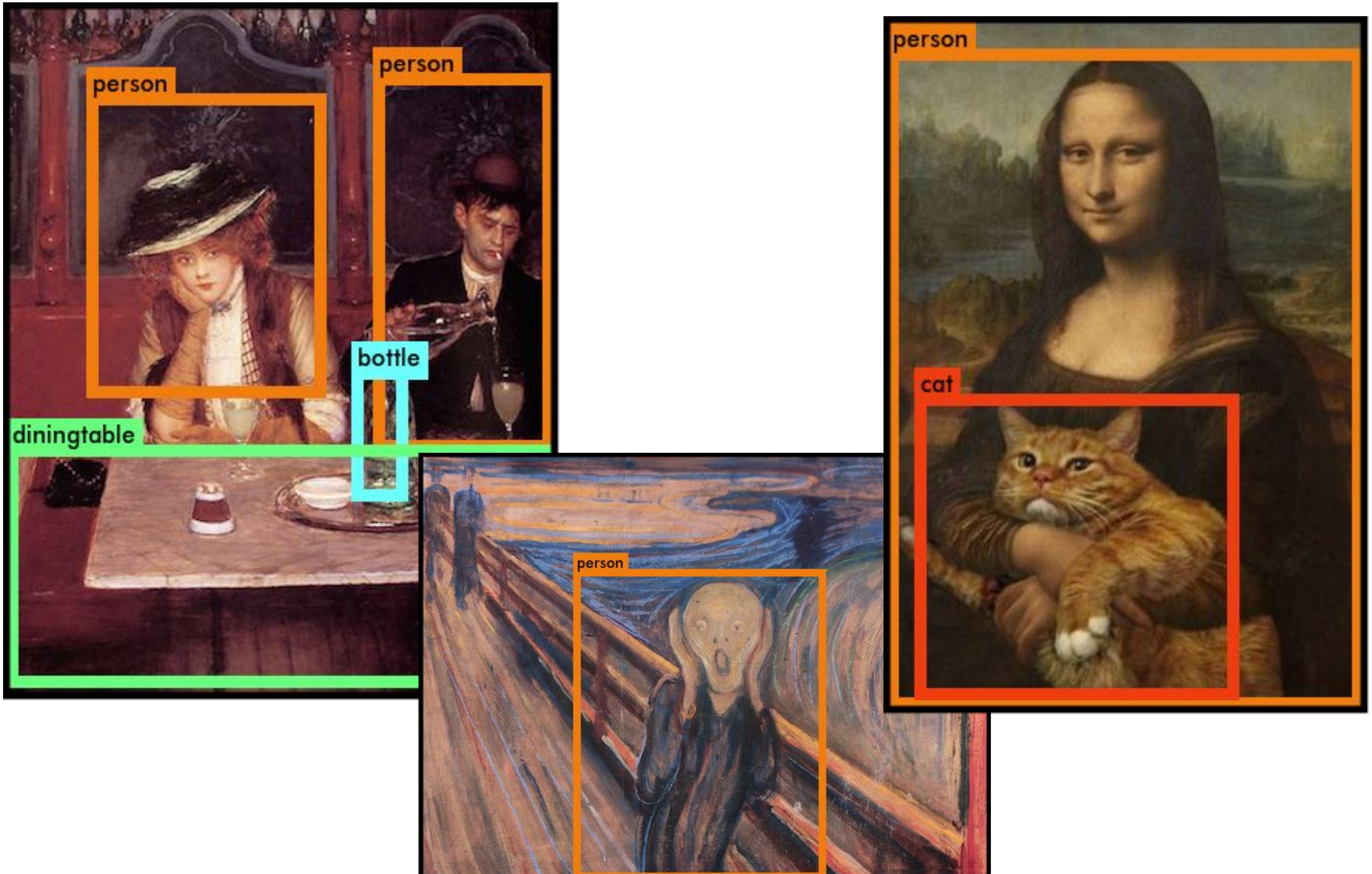
- Pretraining on Imagenet
- SGD with decreasing learning rate
- Extensive data augmentation
- For details, see the paper



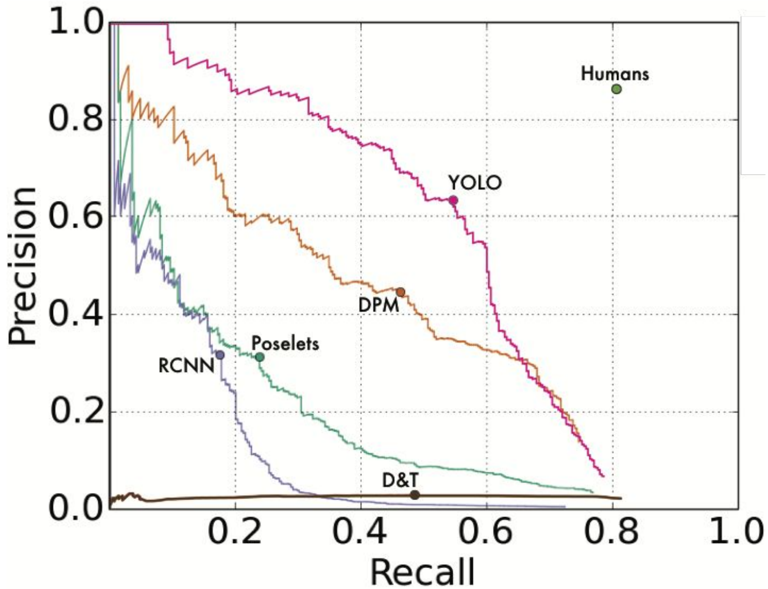
YOLO works across a variety of natural images



It also generalizes well to new domains (like art)



YOLO outperforms methods like DPM and R-CNN when generalizing to person detection in artwork



	VOC 2007	Picasso		People-Art
	AP	AP	Best F_1	AP
YOLO	**59.2**	**53.3**	**0.590**	**45**
R-CNN	54.2	10.4	0.226	26
DPM	43.2	37.8	0.458	32

S. Ginosar, D. Haas, T. Brown, and J. Malik. Detecting people in cubist art. In Computer Vision-ECCV 2014 Workshops, pages 101–116. Springer, 2014.

H. Cai, Q. Wu, T. Corradi, and P. Hall. The cross-depiction problem: Computer vision algorithms for recognising objects in artwork and in photographs.

Code available! pjreddie.com/yolo

	Pascal 2007 mAP	Speed	
DPM v5	33.7	.07 FPS	14 s/img
R-CNN	66.0	.05 FPS	20 s/img
Fast R-CNN	70.0	.5 FPS	2 s/img
Faster R-CNN	73.2	7 FPS	140 ms/img
YOLO	~~63.4~~ 69.0	45 FPS	22 ms/img